APPENDICES

APPENDIX 1 THE ARTICLE PRODUCED DURING THIS RESEARCH AND THE RESPONSE TO IT

Construction of reinforced concrete multi-storey office buildings: a simulation model for time/cost calculations

FIRST STAGE COMPUTER SIMULATION MODEL FOR THE DURATION AND COST ESTIMATION OF THE CONSTRUCTION OF A TYPICAL MULTI-STOREY CONCRETE FRAMED OFFICE BUILDING PROVIDING RAPID RESPONSE TO CHANGES IN INPUT VALUES AND THEIR EFFECT ON THE DURATION AND COST OF THE PROJECT

E. Laptali, N. Bouchlaghem and S. Wild

University of Glamorgan, Department of Civil Engineering and Building, Treforest, Pontypridd, Mid Glamorgan, CF37 1DL, UK

The current model is confined to multi-storey concrete framed office buildings with particular types of foundations and roof and choices within the building envelope, but it has the potential for further development to include wider choice. The flexibility provided by Leonardo will allow, claim the authors, a staged development without affecting the structure of the model.

Le modèle actuel est limité aux immeubles de bureaux à étages, à ossature béton, dotés de certains types de fondations, de toitures et d'éléments d'enveloppe. Mais il présente un intéressant potentiel de développement pour la prise en compte de choix beaucoup plus variés. Les auteurs estiment que la souplesse d'exploitation inhérente au Leonardo permettre un développement phasé sans atteinte à la structure même du modèle

Keywords: concrete frame construction, simulation, time/cost model, UK

Introduction

Duration and cost estimation of construction projects involves handling a number of interacting variables some of which are easily predictable and quantifiable, whereas others rely on intuition and experience. The selection of manpower materials and plant and possible method of work are some of the many factors which affect project duration and cost. The complexity of the problem lies, on the one hand, in the large number of different combinations of activities required to perform a certain task and on the other hand, in the unpredictability of external influences. Furthermore the large amount of data to be handled can render the task unmanageable

0961-3218 © 1995 E &F.N. Spon

Various computer-based time and cost models have been developed since the 1980s. Current effort is directed towards integrated time and cost models where simulation, generation and optimization methods are used [1-3]

Newton [3] discusses the difference between these methods. 'Simulation' presents the structure of a problem where structure refers to how the problem is conceptualized in terms of its boundary, the variables considered and the inter-relationships between variables. A set of input data is provided by the user and then the outcome is evaluated based on a range of other considerations. The 'generation method' produces a set of candidate solutions. This is a more mechanical approach

where for a range of starting values, the model is capable of generating an entire collection of potential solutions. Monte Carlo Simulation is an example of simulation applied to the generation method. The 'Optimization method' evaluates a series of solutions and searches for the best solution for the given criteria.

Each method has its limitations in providing a realistic solution to the practical problems of estimating time and cost, and future refinements are expected to reduce these limitations. One approach to improving effectiveness would be to combine two or more of the existing methods which is a principal feature of the current research project.

A multi-storey reinforced concrete office building has been chosen on which to base the model. It was considered that the disadvantages of construction of *in situ* concrete framed buildings [4], especially being more labour intensive than the other frame types, provide a considerable basis for the application of combination of simulation and optimization techniques.

The duration and cost of construction of a multi- storey reinforced concrete office building is calculated through the simulation model. The model provides a set of choices for the selection of materials and plant and possible methods of work. It also requires the user to input the quantities of work, gang sizes and the quantity of plant required, lag values between activities, output rates, unit costs of plant, labour costs and indirect costs. A linked bar chart is drawn automatically by using the data available from the simulation model.

The optimization component, which is still under development also uses the data provided by the simulation model. The aim is to evaluate sets of solutions, in terms of time vs cost to obtain the optimum time corresponding to the minimum cost under the given schedule restrictions.

This paper presents the structure of the simulation model, the variables, and the procedures considered for duration and cost calculations. Additionally the validation and limitations of the simulation model are discussed.

Data acquisition

The data/information required to be built into the model were accessed from a number of sources. These sources are:

1. Bills of quantities and bar charts of reinforced concrete office buildings which were obtained from two main contractors in the UK.
2. Information acquired during interviews with construction Planners, Estimators and Researchers. Five 2 h interviews were conducted with two Chief Planners. Two 1 h interviews were conducted with two Chief Estimators and the in-house Researchers at one of the contractors provided cost analysis data related to the subcontracting work.
3. Wessex Building Price Book and SMM7.
4. Output rate calculation guide for excavation plant provided by the plant manufacturers.
5. Data provided within published literature.

Table 1. Costs of reinforced concrete multi-storey office buildings

Activity name	Per cent of the total cost (approximate value)
Frame and Cladding, Roof	50
Service	30

The construction activities

For the selection of activities which were to be included in the model, two principal criteria were employed. Firstly the effect of the activity on the overall cost and secondly the effect of the activity on the overall duration of the project.

From a cost perspective, it was decided to include the items which cover 80% of the costs for reinforced concrete multi-storey office buildings. When the cost analysis published by the Building Magazine (Oct,1994, Oct,1993) was combined with the interviews and the research undertaken about cost analysis of sub-contracting by one of the contractors, the following items were identified as summing up approximately 80% of the total cost for reinforced concrete multi storey office buildings (see Table 1).

The literature review [5] and interviews showed that the following should be included in the activity list when duration criteria are considered.

1. Site establishment and clearance.
2. Foundations.
3. Finishes.

Activities (1) and (2) (site establishment and foundations) are the first activities undertaken on a construction site, and impose physical constraints on scheduling. The 'finishes' were stated during the interviews to be the most critical activities for the completion of the project on time.

Model structure

The structure of the simulation model can be divided into three sections. These are the time/cost model, the data files and the bar chart. The user inputs the required information and data into the time/cost model. The input is stored in data files. The stored data are processed by the time cost model and are used for drawing the automated linked bar chart (see Fig. 1).

Time/cost model structure

The time/cost model was divided into two modules: Module A and Module B. This was because two different procedures of duration and cost calculations were employed and activities with the same calculation procedures were grouped into the same module.

In practice duration may be derived by calculation, by quotation from a specialist or by assessment on past experience [6]. In the current programme both duration and cost of construction tasks and activities under Module A are calculated by using

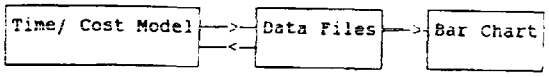


Fig. 1. Information/data flow in the simulation model.

the user input data while both cost and duration of Module B activities are by quotation.

Module A

Module A includes the activities of site establishment and clearance, substructure, superstructure and building envelope. These activities have been presented as the combination of certain construction tasks (see Appendix 1). The information which is built into Module A comprises: resources required for carrying out the activities, output rates of plant and labour, formulae for cost/duration calculations, and a breakdown of the project's main activities into construction tasks.

Module B

Finishes and services are presented in Module B. Due to the detailed and specialist work involved for these two activities the user is asked to input the total duration and cost estimate for the tasks listed in Appendix 2.

Data required for the calculation of cost and duration

For the calculation of project cost and duration the following steps were undertaken.

1. Calculation of direct cost and duration for each task and consequently for each activity under Module A.
2. Calculation of direct and indirect cost, and duration of the entire project by combining results from Module A and Module B.

The formulae used for the calculation of cost (A_c) and duration (A_d) for each task can be found within Appendix 3 and are based on 'Unit Rate Estimating'. Unit Rate Estimating is the calculation of the activity cost, based on the quantity of work and the total time available to perform the task [7]. After calculating the direct cost and duration of tasks under the main activities, the direct cost of the main activities can be calculated by adding each task's cost under that activity. The total cost of an activity includes the indirect costs as well as the direct costs. Indirect costs are those costs which are assessed for the entire project and are not related to individual activity durations. Thus, the total project cost is calculated by adding the direct costs of all activities to the indirect costs.

For the calculation of project duration not only the duration of each task but also the lag values between them must be known.

The preceding discussion and formulae show that in order to calculate the 'cost' and 'duration' of both activities and the whole project, user input is needed for the information concerning the quantity of work, the unit costs of plant/labour and material, the units of plant/labour and material, the lag values and indirect costs.

Resources required for carrying out the activities, related output rates and unit costs

Resources required for carrying out the activities are materials, labour and plant. There are more than 20 screens designed for the input of quantity of materials. These were designed to suit the format of a typical Bill of Quantities (employing SMM7) for ease of input. Alternatively the information could be taken from building drawings, specifications and schedule of rates. The planners interviewed stated that there needed to be more material types to be added to the 'Building Envelope' section. Thus additional input screens are provided for the description of extra materials for cladding. Where a gang of labourers is needed for an activity to be carried out, the user inputs the number of labourers to be employed within that gang.

To minimize user input the Wessex Building Price Book was used as a source of output rates during the development of the model. However, the interviews with the companies indicated that construction firms preferred using their own rates and planners did not wish to employ a model with fixed rates in it. Consequently, although the labour output rates are built into the model, the user also has the option of inputting different values.

Lion [8] stated that building construction is more material and labour intensive than plant intensive. Plant costs represent a comparatively small proportion of construction costs ranging probably around 5%. Although this may be the case for overall building construction, 'site establishment and clearance' is highly plant intensive. Thus it was important to incorporate plant utilization in the model. The range of plant available is however too diverse for full inclusion in the model (refer to British Standard 6031 Earthworks for general descriptions). Wessex Dayworks Plant Guide [10] states that a typical list of plant used in building operations would include JCB 3CX, Hymac 580, Drodt with 4 m^3 bucket, light and heavy vibrating roller and concrete mixer. The plant manufacturers JCB, Wacker and Bormag were contacted and from the information provided the following were included in the model as choices of excavation and vibration plant. Ten backhoes and crawlers with bucket sizes ranging from 0.04 m^3 to 2.25 m^3, and eight vibration rammers and plates with output rates ranging from 150 $m^2 h^{-1}$ to 1050 $m^2 h^{-1}$.

The formula (supplied by JCB) to calculate output rates for excavation plant is:

$$O_e = \frac{E \times 60 \times B_c}{C} \qquad (1)$$

where: E = efficiency factor (dimensionless); O_e = output rate of an excavator ($m^3 h^{-1}$); B_c = bucket capacity (m^3); C = corrected cycle time (h).

The bucket capacities of the excavation plant are provided by the manufacturers. The user can see these on the input screen while choosing the excavation plant. This allows the user to choose the machine that fits most closely to requirements

To determine the plant output rate it is necessary to determine the corrected cycle time from the basic cycle time.

Basic cycle time is the time taken for an excavation plant to load, manoeuvre, dump and return to dig. However, in working conditions the cycle time is not equal to the sum of these actions as the performance of excavation plant is affected by various factors. These are depth of the excavation, type of material to be excavated, type of the target into which the excavated material will be unloaded, slew angle of the excavator, ground conditions, and the status of the operator. Thus, corrected cycle time is calculated by adding the compensation figures for the above factors to the basic cycle time of the particular excavator. Additional to the above factors that affect the cycle time, there may be some interruptions due to operator breaks, maintenance and job holdups. The effect of these factors are accounted for by the efficiency factor (E). All the factors and basic cycle time for each excavator were provided by the JCB manufacturer.

Additional to the input of resources required to be used and output rates corresponding to these resources, the unit cost of the resources are also asked to be input by the user.

The precedence relationships

The precedence relationships of activities are dependent on the constraints on scheduling. Echeverry and colleagues [10] divides these constraints into four groups. Physical relationships among building components, trade interactions, path interferences, and code regulations.

The sequence of works are well defined in repetitive construction processes [11]. Birrell [12] emphasizes that multi-storey office building construction requires sequential waves of work especially for the shell. More specifically Kartham and Levitt [14] state that the physical relationships themselves can be used to generate much of the needed sequence logic for multi-storey office building projects. However, it should be noted that for the cases of external and internal cladding, finishes and services there can not be a particular activity that can be generalized as being the preceding task. Therefore, it is assumed that these activities can start any time preceding staircases from ground to first floor. Thus, the user has to specify the floor of the staircases that will be succeeded by the external and internal cladding, finishes and services.

The precedence relationships between activities require that before an activity can start, all the activities that precede it, should be partially or wholly completed. For this, four types of precedence relationships can be established between two sequential tasks. These are finish-to-start, start-to-finish, start-to-start, and finish-to-finish dependencies where each can include a lag value. Bar charts reported in the literature and those used by the participating construction firms showed that start-to-start and finish-to-start precedence relationships are the most widely used sequencing. Based on the above discussion and the validation of the programme, start-to-start precedence with lag values was incorporated in the model although finish to start relationship can also be easily adopted.

Indirect costs

There have been different approaches to what constitute the indirect costs [14,15]. The items included within this model are the ones which are stated under the 'Project Overheads Schedule' in Code of Estimating Practice [7].

Output and validation

Output is displayed in two forms; output screens and linked bar chart. An output screen for each activity displays the duration and direct cost calculated for each construction task included under that activity. There is also an output screen displaying total project cost and duration. The linked bar chart is produced utilizing the data on duration of activities, lag values between activities, and number of floors.

The time and cost model which utilizes the Leonardo knowledge based system and the linked bar chart which has been developed using QBasic programming language are automatically linked to each other enabling the bar chart to be called from the Leonardo system. Exiting the bar chart brings the user back to the time and cost model.

The validation of the results obtained from the model was undertaken in two stages, validation of the duration values, and validation of cost values. A bill of quantities for a hypothetical six-storey reinforced concrete office block was prepared. The validation of the duration values was achieved by comparing the bar charts produced by two Planners from the participating companies for the

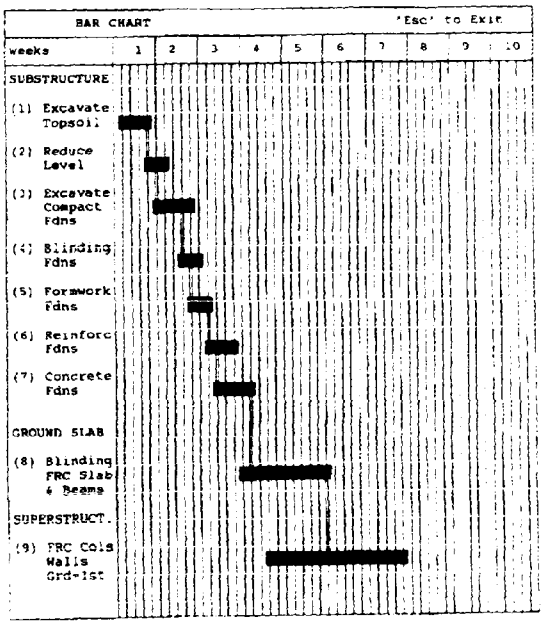


Fig. 2. Display of the linked bar chart.

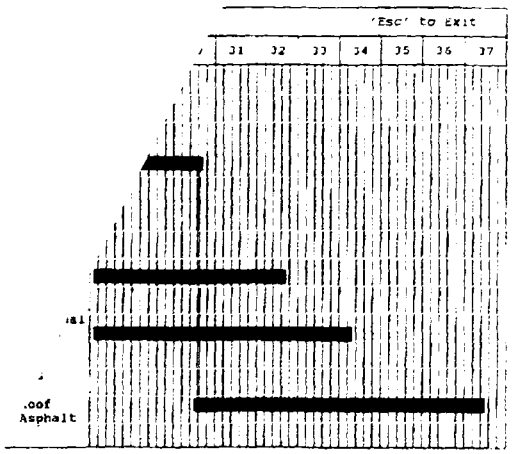


Fig. 3. Display of the linked bar chart.

hypothetical building with the bar chart produced from the computer model. The model gave exactly the same duration results when the data used by the planners were input into the model. Figure 2 and Fig. 3 illustrate the first and the last screen displays for the linked bar chart drawn for the hypothetical building mentioned above. Figure 4 shows the linked bar chart drawn by one of the planners. As the estimators were not able to use their own unit cost data, hypothetical costs were employed and the validation of cost values was achieved by comparing manually calculated costs with those produced by the computer model. Good agreement was achieved.

Limitations and future developments of the simulation model

The current limitations of the simulation model can be stated as follows:

1. The simulation model is developed for multi-storey (any building with more than two floors) buildings. The time/cost model can be developed for other frame types by adding information in the 'Superstructure' section. However, a new bar chart has to be developed for the new frame type.
2. The 'uniqueness' of every construction project may require additional activities to be included. For example the choice of only one type of foundations and roof, and the limited types of materials can be restrictive for some projects. However modules can be developed for calculation of duration and cost of these additional activities within the model.

The benefits obtained by the model

The benefits of employing the simulation model can be stated as follows:

1. Output from the model is in an easily understandable form (linked bar chart) instead of presentation in figures.

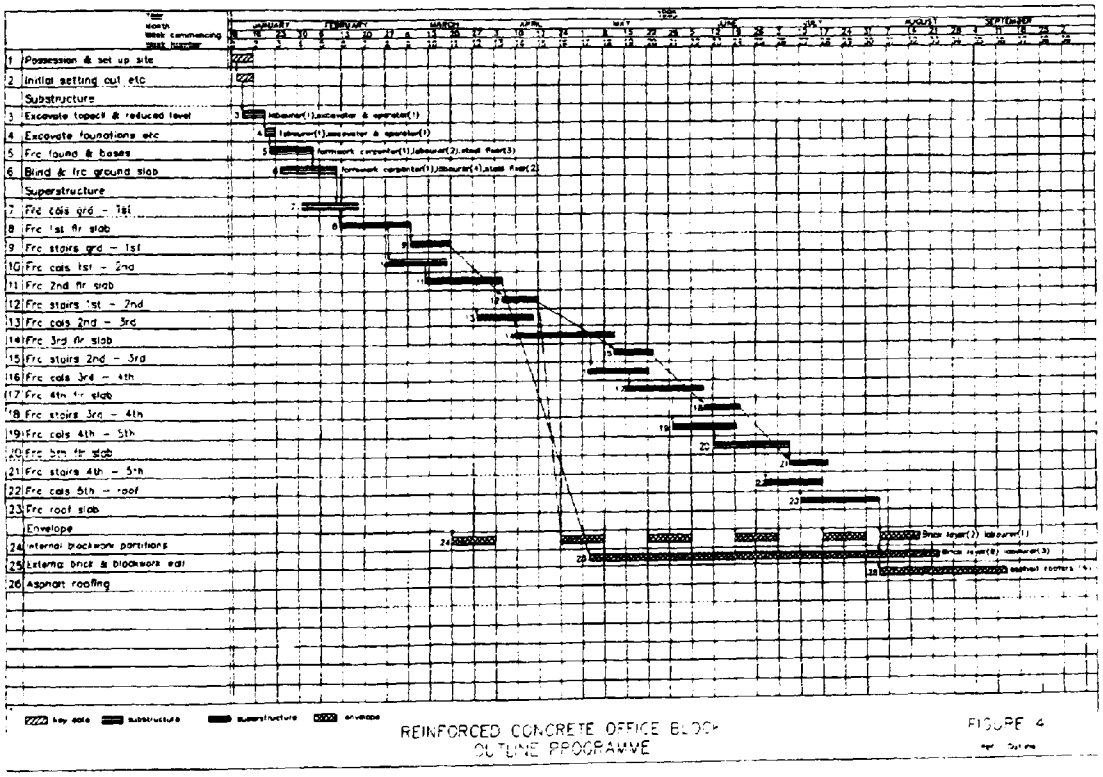


Fig. 4. The linked bar chart drawn by a planner for the hypothetical building.

2. The effect on duration and cost, of changes in output rates, gang sizes and other input values, can be rapidly observed. The effect is also automatically displayed in the linked bar chart.
3. The data input for a project can be used or updated for subsequent projects as all user input is stored into data files and called back by the model.
4. The simulation model can be used as a teaching or training aid which would help students be familiar with SMM7, bill of quantities, and the interaction between some of the procedures for planning and estimating.

Conclusion

A simulation model for the duration and cost estimation of the construction of a typical multi-storey concrete framed building has been presented. This is part of an integrated model which will evaluate sets of time vs cost solutions and obtain the optimum time corresponding to the minimum cost under the given schedule restrictions. The duration and cost of a whole project is calculated within the simulation model and a linked bar chart is automatically drawn to present the duration values.

The model produced exactly the same duration and cost results as those manually provided by the Main Contractors, when the same input values were used (ie. output rates, unit costs, lag values, gang sizes) which establishes the reliability of the model. The model provides rapid response to changes in input values and their effect on the duration and cost of the project. Also the effect is automatically displayed in the linked bar chart. It also allows input to be stored in data files for future use on new projects.

The current model is confined to multi-storey concrete framed office buildings with particular types of foundations and roof and choices within the building envelope, but it has the potential for further development to include wider choices. In addition a future development will be to build information into Module B for calculation of duration and cost in this module. The flexibility provided by Leonardo will allow a staged development without affecting the structure of the model.

Acknowledgements

The authors would like to thank the University of Glamorgan for providing funding for a research assistanship to carry out this research and Professor P. S. Coupe for providing the facilities in his department.

References

1. Cusack, M. (1984) The Use and Limitations of Mathemanical Models in the Planning and Control of Construction Projects. *Construction Management and Economics*, **2**, 219–24.
2. Bennett, J. and Ferry, D. (1987) Towards A Simulated Model of the Total Construction Process. *Building Cost Modelling and Computers*, 377–85.
3. Newton, S. (1991) An Agenda For Cost Modelling Research, *Construction Management and Economics*, **9**, 97–112.
4. Illingworth, J.R. (1993) *Construction Methods and Planning*, 1st edn., E. & F.N. Spon, London.
5. Ashworth, A. and Skitmore, R. M. (1990) Accuracy in Estimating, Occasional Paper, No.27. CIOB.
6. CIOB Report (1991).
7. Code of Estimating Practice (1983) CIOB Publications.
8. Lion, E. (1980) *A Practical Guide to Building Construction*, Prentice Hall, London.
9. *Wessex Dayworks Plant Guide 1992/93*, 1st edn., Wessex Publishing Ltd.
10. Echevery, D., Ibbs, W. and Kim, S. (1991) Sequencing Knowledge For Construction Scheduling, *Journal of Construction Engineering and Management*, **117**, 118–30.
11. Baldwin, A. and Thorpe, A. (1990) Information Technology For Construction Cost Estimating, *Habitat International*, **14**, 157–63.
12. Birrell, G.S. (1991) Factors and Variables to Induce the Effective Construction Schedules and Efficient Construction Management. In *Quality and Economics in Building*, 1st ed., pp. 34–42, E. & F.N. Spon, London.
13. Kartam, N. and Levitt, R.E. (1990) Intelligent Planning of Construction Projects, *Journal of Computing in Civil Engineering*, **4**, 155–75.
14. Scott, D. and Kagiri, M. (1992) Choosing the Most Appropriate Method of Construction with Computer Assistance, *Construction Management and Economics*, **10**, 153–77.
15. Tah, J.H.M., Thorpe, A. and McCaffer, R. (1994) A Survey of Indirect Cost Estimating in Practice, *Construction Management and Economics*, **12**, 31–6.

Appendix 1. Activities under Module A

1. Site establishment and clearance:
   (a) Excavation of topsoil to be removed/preserved.
   (b) Excavation to reduce levels.
   (c) Excavation for basements.
   (d) Excavation for trenches.
   (e) Compaction of bottom of excavations.
2. Substructure: (reinforced) concrete foundations
   (a) Formwork erection.
   (b) Reinforcement construction.
   (c) Plain/reinforced concrete placement.
3. Superstructure:
   (a) Formwork erection.
   (b) Reinforcement construction.
   (c) Plain/reinforced concrete placement.
4. Building envelope:
   (a) Brickwork and blockwork construction.
   (b) Roofing, reinforced concrete roof structure:
      (i) Formwork erection.
      (ii) Reinforcement construction.
      (iii) Plain/reinforced concrete placement.
      (iv) Roof asphalting.

Appendix 2. Activities under Module B

1 Finishes:

(a) Internal finishes

(i) Wall finishes
(ii) Floor finishes.
(iii) Ceiling finishes.

(b) External finishes.

2 Services:

(a) Lifts.
(b) Mechanical services
(c) Electrical services.

Appendix 3. Formulae used for cost and duration calculations

$$A_c = U_{cl}A_dN + Q_wU_{cm}$$

$$A_d = \frac{Q_w}{O_r U}$$

where: A_d = Duration of a construction task; Ac = Direct cost of a construction task; Q_w = Quantity of work (q); O_r = Output rate of particular equipment (labour per hour); U = Units required; U_{cl} = Labour/plant unit cost (£ per h); N = number of labourers employed; U_{cm} = Material unit cost (£ per q); q = Unit of quantity (m, m^2, m^3, ton); U = Units required.

REINFORCED CONCRETE

Reinforced Concrete Council
Century House, Telford Avenue,
Crowthorne, Berks RG45 6YS
Telephone: (01344) 762676
Fax: (01344) 761214

28 September 1995

Messrs Capati, Bouchlagham & Wild
University of Glamorgan
Dept of Civil Engineering & Building
Treforest
Pontypridd
Mid Glamorgan
CF37 1DL

RCC

Our Ref: CHG/1184/KJ

Dear Sirs

I was very interested in your article in Building Research and Information regarding a simulation model for time/cost calculations.

The RCC has been involved with trying to optimise different forms of concrete construction for the buildings in our Cost Model Study. We have designed different forms of construction eg:

Ribbed slabs
Flat slabs using traditional loose bar with traditional links
Flat slabs using one-way mats and proprietary shear systems.

We have designed these at different thicknesses and have estimated material cost differences. We have looked at theoretical changes to the engineers fees due to degree of difficulty.

However, we have had great difficulty in estimating critical construction time differences for the different forms of construction. Are you in a position to help?

We are part-way through a DOE PiT project looking at rationalisation of reinforcement in flat slabs in the run up to the construction of the in-situ concrete frame test facility at BRE Cardington early next year. Perhaps there would be areas of mutual benefit and I would look forward to hearing from you.

Yours sincerely

Charles Goodchild

Encl. Cost Model Study

APPENDIX 2 INTERVIEWS WITH THE CONSTRUCTION PLANNERS, ESTIMATORS AND RESEARCHERS

Introduction

During the development of the integrated model, various interviews have been undertaken by construction practitioners (planners, estimators and teaching company associates) from the companies Wimpey and Kyle Stewart Construction.

The interviews have been in structured form and undertaken during different stages of the model. In general the aim of the interviews have been;

(1) to get information/data that could be used for the development of the model,

(2) to get the opinions of the practitioners related to the model and its potential application areas,

(3) to validate the results from the model.

Interview No. 1

At : Wimpey Construction, Bristol

Date : 28/2/94

Aim: To get to know the company and introduce the aim of the project.

The first interview with the collaborating establishment was in the form of getting to know each other. Firstly, the aim of the project was explained to the two planners and an estimator from Wimpey and then the practitioners gave, in general terms, some information about the firm and their job content.

Interview No. 2

At: Wimpey Construction, Bristol.

Date : 10/5/94

Aim: To acquire information about estimating and planning procedures and to obtain the opinion of the planner about the computer model for some more improvements on the model.

First some general questions were put to the planner about the estimating and planning processes and their interaction. After a discussion of these questions, the computer model was presented to the planner, and some additional questions were asked about the model.

Section 1: General Estimating and Programming Questions.

Q(1.1): How do you (or Wimpey Construction Estimating and Planning) define the stages of a construction project?

(a) Pre-tender - Tender - Pre-contract - Contract

(b) Pre-contract - Contract - Post-contract

A(1.1): (a) The definition of stages at (a) was employed by Wimpey. The planner defined the pre-tender stage as the stage where the client listed the contractors that could go for the tender from a starting list of many contractors.

To be selected to the tender list, the contractor has to answer a set of questions of the client in a report. The questions are mainly about:

- information about previous jobs,
- qualifications of staff,
- Health & Safety policies,
- Company structure,
- Quality assurance,
- Organisation charts

The planner complained that every client gave different questions to be answered and this took a lot of time and money. He wished that all of the clients asked the same questions.

The planner then added that after giving the report to the client, some of the clients required an interview with the project team (planner, estimator and project manager) to get a feeling for the company. However, it was stated by the planner that the decisions taken after these interviews were very subjective, and they involved no scientific way of choosing the contractor.

It was stated by the planner that the progress reports during the contract stage were given to the client on a monthly basis, and to the project manager on a weekly basis. (by the site office)

Q(1.2): How much of the information used in the estimating process is required in the planning process during the tender stage (traditional procurement)?

A(1.2): The planner answered this question by saying "nearly a 100 percent". Then he added that the Bill of Quantities was used by both departments, however while the estimators priced every item in a bill, the planners summarised the items under main headings and then used this information to produce a programme.

He also stated that both departments worked very closely together, and to give an example, they worked together in deciding the gang sizes and indirect costs.

Q(1.3): What are the main characteristics that affect the degree of accuracy of an estimate and a plan?

- Type/sophistication of project. (Industrial/Commercial)
- Time to complete tender.
- In house resources available.
- Importance given by company to that particular project/tender.
- Number of packages to be sub-contracted/estimated.
- Amount of work to be completed/estimated my Main Contractor's operatives.
- Accuracy of sub contractor's estimate.
- Fluctuations in cost, clients variations.

A(1.3): The planner stated that they never checked the accuracy of an estimate when a project was completed. The accuracy of an estimate was checked only if they lost a lot of money on the job. He also added that they did not have any allowable margin for the accuracy of an estimate, but it should be very accurate especially in time of recession.

He also added that the estimator's work for a particular project finished after the tender stage and QSs on site took the responsibility of checking the costs.

However, he still made comments on the following points which might affect the accuracy of an estimate.

(1) If time to complete tender is short the accuracy is affected negatively.

(2) More people working on the estimate, estimate is more accurate.

(3) More importance given to the tender, more accurate is the estimate.

At this point, he stated that the biggest overhead cost in a construction firm was the tendering team (planning, estimating), and the tendering team was paid from the profit on site. On the other hand the site staff are counted within the cost of the site.

He also added that in the current year they had a tender success ratio of 1 in 6.

(4) Fluctuations in cost affect the estimated price. On some jobs (ex: a hospital job they waited to built for 2 years) where there are a lot of clients' variations, the estimate is affected by these variations.

(5) The sub-contractors' estimates affect the tender price, and the planner stated that there might be up to 40-50% difference between the estimates of different sub-contractors.

He also stated that if a sub-contractor was busy, the estimate from that sub-contractor was usually high.

The planner stated that Wimpey subcontracts all work, and also stated that his company prices the main activities of groundwork, brickwork, joints, carpentry, concrete, formwork, reinforcement to check the estimates from the subcontractors.

(6) Most companies preferred front end loading, as the cost of the project at the beginning was high due to the high cost of site set up and foundations.

If back end loading was preferred then the company had to borrow money from a bank and this affected the estimated cost.

He said that if there was front end loading, or back end loading for the project it was stated in the bill of quantities (in the preliminaries). Thus the estimate was done according to that.

Q(1.4): What type of structure, steel or concrete, is more commonly found in office building projects?

A(1.4): The planner stated that one could not generalise that concrete or steel was more common. It all depended on the job. Concrete provided the flexibility where steel provided speed. However, he also pointed out that steel structures were more preferred within the UK construction industry.

Section 2: Questions Relating to the Computer Model.

Q(2.1): Can the Pareto distribution (20/80 : i.e.; 20 % of work accounts for 80% of cost) be applied to the activities included within the programme for a Commercial project? If not, what is the percentage of those activities included for the development of an office building.

A(2.1): The planner gave the following percentages of cost.

10 % Foundations	50 % RC Frame, Roof, External cladding
10 % Finishes	30 % Mechanical

Thus, the activities included within the Simulation Model account for nearly 60 % of the cost of the work.

Q(2.2): Can you say that the activities used within the computer model are the critical activities for office building development?

A(2.2): The planner stated that the activities were the critical ones but there needed to be finishes at the end. He also said that in some projects finishes affected the duration more than for others.

Q(2.3): Do you believe the currently completed part of the programme has any use in industry?

(a) Yes, why?

(b) No, why not?

A(2.3): The planner answered this question by a " Yes". He then pointed out that although the computer model was restricted to certain activities, it was a good guideline and it could be used as a training tool within the planning and estimating departments.

Q(2.4): Do you believe that the programme would help students to understand the following processes more easily:-

(a) the use and format of bills of quantities (SMM7),

(b) the estimating and planning processes,

(c) the interdependence and interaction of the estimating and planning processes and disciplines?

A(2.4): The planner thought, the computer model would be a very useful tool to teach the above points to students. He also added that most students leave university without seeing any bill of quantities.

He also stated that estimating and planning departments worked very closely with each other and it was very important for the students to learn that.

Q(2.5): Is the information asked for during the execution of the programme available during the tender stage?

A(2.5): "Yes". Then the planner added that the procedure employed in the programme was the same procedure employed by the planning and estimating departments during the tender stage.

Q(2.6): Do you (Wimpey) take into account the loss in productivity due to the increase in the number of operatives/labour?

A(2.6): The planner stated that they did take productivity loss into account due to an increase in the number of operatives, however there was no scientific formula used for it. He said that a good estimate was needed for it. He also pointed out that there were a lot of variables that had an effect on the productivity like the effect of weather, complexity of the job, age of the operative, overtime work.

(a) Is productivity affected by shift work?

The planner answered "yes", as they had to work under none natural lightning.

(b) Is productivity affected by over time?

The planner answered "yes" as they get tired.

Section 3: Usage of Programme.

Q(3.1): What part of the programme can be improved to make it more user friendly ?

A(3.1): The planner stated that the programme would be more user friendly if it was possible to see the quantities of work and the output rates together and also by being able to print them.

Section 4: Any Other Comments.

(1) The planner stressed the point that planning could not be standardised as every project was unique and there were so many different work combinations for a project.

(2) The planner stated that Wessex and Spons were very generous with their output rates and unit prices, but they provided good guidelines.

Interview No. 3 :

At : Kyle Stewart , London

Date : 21/6/94

Aim: Getting information about the contribution of different sub-contracting packages on project cost.

The researchers (Teaching Company Associates) provided some data about the percentage (%) of main sub-contracting packages in a contract price. The findings were achieved through a teaching company programme with Loughborough University of Technology. The following table presents their findings.

SUBCONTRACT PACKAGES

Percentage (%) of Total Contract Price (Traditional & Design and Construct:

	Minimum %	Maximum %
Electrical :	6.4	31.1
Cladding	5	28.2
Mechanical	5.5	27
Frame	2.7	26.8
Ground Works	4	19

Interview No. 4 :

At : Wimpey Construction, Bristol

Date : 4/7/94

Aim: Validation of the Results Obtained by Running the Computer Model

The aim of this interview was to create a hypothetical reinforced concrete office building, prepare the bill of quantities for it and finally calculate the duration and cost of construction.

Before the interview a bill (of quantities) was prepared which included the item names that have been built into the simulation model.

During the interview, the planner drew the plan, the elevation and some sections which belonged to a hypothetical concrete framed office building. (See Appendix 5 for the drawings)

The reason for using a hypothetical building was to cover items identical to those in the computer model so that all parts would be validated. Also the planner was not able to authorise the loan of any bills of quantities held by the company.

During the preparation of the bill of quantities, the planner suggested leaving out some items and adding others to make the model more suitable for reinforced concrete office blocks. These were:

(1) Take out the "stainless steel for reinforcement" as stainless steel was not used in office buildings for reinforcement.

(2) Add brickwork and blockwork for external walls, in addition to internal walls.

(3) Add asphalt for roofing.

The planner calculated the duration for carrying out each activity by using their own output rates and gang sizes.

It was agreed that the next step would be to input the values into the computer model and compare the final duration values with the values obtained by the planner. Then in the next interview, the differences between the results could be discussed and also a bar chart could be drawn.

It should be noted here that the same bill of quantities for the hypothetical building was given to the planner in Kyle Stewart and a bar chart was drawn by that planner, too.

Interview No. 5 :

At: Wimpey Construction, Bristol

Date: 25/7/94

Aim: The Validation of the Results, Obtained by Running the Computer Model

The aim of this interview was to continue working on the calculations of duration of activities of the hypothetical building which was designed in the previous interview.

The planner had finished the duration calculations and drew a bar chart for these by using PowerProject programming software. The planner and the researcher discussed the following matters relating to the bar chart.

The Main Differences Between the Planner's Bar Chart and the Bar Chart Drawn by the Computer Model.

(1) The planner put the brickwork headings after all the concrete work was finished. The researcher stated that in her programme the bar chart showed the brickwork after each floor. The planner stated that this was not an important difference but it is usually preferred by the planners to "keep like items together in a programme". He also stated that generally brickwork started after at least 3 floors of concrete work had been finished.

(2) The researcher also realised that the planner put the RC to stairs on the same line for all floors, not after each floor as the researcher did in her programme. The planner also said it would not matter if you put all the RC for stairs together or in each floor separately as RC for stairs was not critical and could finish any time before the end of the project.

(3) The planner stated that there was usually 3 weeks lead between the start times of concrete work (to columns) to two floors.

(4) The planner stated that although the duration of concrete work was the same for all of the floors on the bar chart, it was better to employ a learning curve for the labour in a job like this (repetitive construction).

Time and Cost Relationship:

(1) The planner stated that the most important item in terms of time and cost relationship was formwork because if the completion time of the project was extended the formwork could be used over and over again which would decrease the cost of formwork and eventually the direct cost of the project.

(2) Direct cost calculations: The planner stated that it was a good decision in the computer model to ask the unit cost/price of the resources from the user, as the unit cost/price would fluctuate from time to time and place to place and would depend on the availability of the resources.

(3) Indirect cost calculations: The planner stated that after the calculation of the duration of the project the cost preliminaries would be calculated accordingly. He stated the following main headings would be employed for most of the jobs.

(a) Cost of supervision.

(b) Hutting on site.

(c) Erect and dismantle the huts.

(d) Compound areas for huts to stand on.

(e) Security.

(f) Hoarding and temporary fencing.

(g) Notice board.

(h) Telephone instalment and rental.

(i) Office furniture and equipment.

(j) Plant used (Plant schedule).

(k) Erect and dismantle plant

(l) Scaffolding

(m) Transport of offices/plant

(n) Sundries:

(n1) Samples

(n2) Testing

(n3) Protection

(n4) Survey Equipment

(n5) Printing drawings

(o) Cleaning the building.

(p) Drying out the building.

(q) Attendance on nominated sub-contractor.

(r) Temporary facilities:

(r1) Water

(r2) Electricity

(r3) Drainage

(r4) Access road

(s) Temporary works.

He finally stated that the preliminaries usually add up to about 8-15 % of the direct cost for the project.

The Relationships Between the Activities:

The planner stated that at the tender stage the start relationship between the activities were more important for the project duration calculations.

Interviews No 6 & 7 :

At: Kyle Stewart Ltd., London, Wimpey Construction, Bristol.

Dates: 13/1/95, 20/1/95

Aim: To learn about the estimating procedure from the estimators.

The following are a combination of the answers from both of the estimators.

Q(1): What does the estimating procedure include?

A(1): It includes mainly the unit pricing of the contract bills. This consists of all the preparation required before pricing, drawing up and sending out of enquiries for various materials or works within the bill of quantities. The person responsible for the enquiries should be familiar with the sources of supply for materials and with the sub-contractors available in the area where the project is to be undertaken. This is usually undertaken by the co-operative work of the estimating and the buying departments.

Q(2): What are the main elements included in the unit rate estimating?

A(2): These are mainly labour and materials and additionally plant.

Q(3): What do you consider as the unit labour cost for the labour working in a gang with different skill levels ?

A(3): Firstly, the weekly cost, i.e. wage, of a labourer is calculated by multiplying the hourly rate by the total number of hours expected to be worked each week. The hourly rate includes the basic hourly pay plus minimum bonus, plus additions to the basic hourly rate paid as attraction, travelling time, expenses and subsistence allowances, employers' and public liability insurance, tool money, sickness benefit payments and holidays with pay scheme. Additionally national insurance and pensions; and guaranteed week allowances which are expressed as a percent of the net wages paid.

The net hourly labour cost is calculated by dividing the total weekly cost by the actual hours to be worked. The gang rates are then calculated by considering the proportion of skilled, unskilled and semi-skilled labour that work together in a gang.

Q(4): How do you consider the material unit costs?

(A4): These are requested from the suppliers or manufacturers by quotation. In this way, it is ensured that the unit rates are up to the current market rate. The unit rates should include mainly the basic price, less discounts other than those for cash,

allowances for waste, unloading, stocking, getting in, and distributing around site. Including all these factors provides a more accurate material cost.

Q(4): How do you consider the plant costs?

A(4): It depends on the use of the plant. If a particular plant is required for a particular task, the cost of it is calculated by the unit rate estimating method. However, the type of equipment and plant which has various uses (e.g. a tower crane) or is used by different trades is included under 'preliminaries' as a bulk cost.

Q(5): What are the advantages of using unit rates representing costs only, i.e. not including overheads and profit.

A(5): Using unit rates representing costs only has two main advantages:

(1) If a percentage is added to each unit rate, rounding-up the figures may be required (to the nearest 1p or 0.5p). This may add a considerable amount to the total estimate for some very large projects and may result in loosing the job because of too high a tender price.

(2) If an overhead or profit element is added after all the unit rates are completed, it will not be easy to make adjustments to all the items.

Q(6): Do you use any computer packages for estimating?

A(6): Yes. Mainly as an aid to cut out the repetitive calculations for similar or common items. However, these are only an aid during the estimating procedures as 'estimating is an art not a science' and some procedures cannot be formulated.

Q(7): Do you work closely with the planners?

A(7): Sometimes. The contract duration is used while calculating the cost of time based items under preliminaries, i.e. the indirect costs. Thus, calculation of a realistic duration is very important for a realistic estimate. If the estimated contract duration is too short, the time based items will be undervalued and it may cause a loss for the contract. If it is too long, it may mean a loss of competitiveness due to too high a cost for the time based items.

Q(8): How do you account for the productivity loss which results from an increase in the amount of labour in a gang to accelerate work?

A(8): (Wimpey) We use 3 labourers as the maximum number of labourers to be a 100% productive in a gang. However, this number may be different for other estimators in other companies.

Interview 8 :

At: Wimpey Construction, Bristol

Date: 20/1/95

Aim: To obtain the opinions of the planners on optimisation applications.

(Q1) Do you always plan for a given project duration?

(A1) "Not always. For 75% of projects the client gives a project duration which is required. At the same time, the client normally asks about the benefits of working to a duration given by the company. Experienced clients (like Sainsburys asking for the construction of a superstore) usually know how long it takes to finish a project. This duration normally only changes with different ground conditions."

(Q2) Do you employ any optimisation technique during pre-tender/pre-contract stages?

(A2) "No. But we use 'what if' situations for the use of resources and choose the resources that would satisfy the requirements of the client. We usually know what resources are available on the market. To give an example, if there are two to three big reinforced concrete frame building construction projects in progress at one time in Bristol, they would roughly know the number of men available in Bristol to carry out a new

reinforced concrete frame building. If the resources are not available in that particular area, they look further afield to other areas. Depending on the availability of resources the market rates may change accordingly. Thus they try to achieve tender values as 'optimum' as possible for the conditions of that particular project. This is one form of optimisation which they employ during pre-tender and pre-contract stages which is somewhat empirical."

(Q3) To be able to finish a project in a shorter duration than calculated or required, would you employ more resources (plant and labour) or overtime work?

If you employ more resources, do you make allowances for the productivity loss due to overcrowding and congestion?

If you employ overtime work, how do you account for the cost increase?

(A3) Although at times more resources are employed to accelerate the work it should be noted that the duration does not decrease in direct proportion to the resource increase. For example if it takes 4 men to build a roof in 6 weeks, on doubling the work force to eight men it will take longer than 3 weeks to construct the roof. This is because efficiently decreases with the larger numbers of workers due to increased congestion. There is no

formula or scientific procedure which can be used to predict the exact outcome and it is left to quotations by the subcontractors to determine the cost.

In cases where the duration required by the client for the completion of the project is too short, the contractor would suggest that it would be cheaper to build over a longer period. However the client has to take cash flow and the funding situation into consideration for a longer duration.

(Q4) Are there any cases where the total indirect cost of a project is greater than total direct cost?

(A4) No. Not even in very specialised work like the second Severn Crossing. Even in that project the indirect costs would be about 20% of the direct costs. Usually it is 10-12%.

(Q5) Do you take resource limitations into account during pre-tender and pre-contract planning?

(A5) No. If the company says it cannot provide the resources, it will not get the job. It is always assumed there are resources available. The availability of resources is all a matter of supply and demand. In some cases getting the resources is not only achieved by paying the market rate but also including

additional attractions such as bonus payments or paying at greater than the market rate.

(Q6) Can we say that total indirect cost is equal to 0 when the construction of the project has just started (i.e. project duration is 0)?

(A6) Yes, that can be assumed for traditional procurement.

(Q7) What is your opinion of optimisation applications?

(A7) At tender when you put the tender price, the cost of the project is only an estimate. Only during and after the construction of the building do you discover whether the values were optimum or not. If the project falls behind schedule and profit falls it shows that your tender estimates of time and cost were not the optimum.

Interviews No 9 & 10 :

At: Wimpey Construction, Bristol, Kyle Steward Ltd., London.

Dates: 10/10/95, 17/10/95

Aim: The Opinions of the Planners on the Model

After the completion and the validation of the model, the opinions of the three planners (two from Wimpey and one from KS) were sought to give their assessment of the model. Their opinions can be summarised as follows:

(1) In its current form, the model could be used for training within large construction firms and also for estimating and planning by medium sized contractors. Additionally, competition within the construction industry requires firms to tender for as many contracts as is possible. Thus, this model could be used to help less experienced planners and estimators to prepare some of the many tenders.

(2) Any future development of the model should include steel frame structures as steel frames are currently more widely used in the UK.

(3) While deciding on purchasing such a model construction firms would give greatest emphasis to two criteria. These are, the accuracy of the results, and the purchasing/operating cost benefit ratio.

(4) The use of only physical constraints on the lag values are found to be sufficient for repetitive work.

(5) The detail of the model is ideal for the pre-tender and pre-contract planning stages.

(6) The detail in the output presentation is sufficient enough for different management levels. However, it would be an advantage to display the costs of labour, material and plant separately.

(7) It was stated that although accelerating the project is not usually considered during the tendering procedure, there have been cases where optimisation has been applied as an approach in extending the planned project duration to meet the requirements of the client.

(8) It was stated that the linked bar chart is currently the only programming technique that is used by the practitioners. Although the linked bar chart drawn by the model is stated to be sufficient for the current development, it would require modification if additional new activities were to be included within the model. It was also stated that clients sometimes ask for programmes and related information to be provided a floppy disk and thus they prefer the tender programme to be produced using the

computer software that is available to them. Thus, the model would be accepted by the industry if it could also interact with the currently employed bar chart drawing computer programmes.

(9) It was stated that neither of the firms utilise any computer model which combines both cost and duration estimation. However, use of such a programme would help to change the structure of the industry by enabling the estimators and planners to work closely together.

(10) For the effects of uncertainty on cost and duration estimates and intuition are used more than quantitative methods. However, these are taken into account within the output rates. Use of probabilistic models based on techniques like Monte Carlo Simulation are too sophisticated to be accepted by the industry.

(11) It would be an advantage for the model to provide cash flow curves as the use of S curves has been increasing in the industry especially after the introduction of the Private Finance Initiative.

(12) While assuming a linear relationship between activity cost and duration, the gang sizes are assumed to be increasing one man at a time. However, for some trades (like steel

fixers, brickmen and carpenters) the gang sizes would not be increased by one man only as they work in groups of two or more. Thus, an incremental increase in gang sizes would be more realistic.

(13) It was stated by one of the planners that the time taken for reinforced concrete to achieve its required strengths affects the duration and cost of the project due to the different curing times needed. The model could be used to compare the effect on project duration and cost of different curing times and strengths of the reinforced concrete.

(14) One of the planners insisted that the future development of the model should consider the development of the 'Superstructure' (reinforced concrete) section in more detail which would provide a 'Reinforced Concrete Frame Detailed Contract Programme'. He stated that this would mean considering additional aspect to the model like concreting of the floor slabs by dividing them into sections. The possibility of checking the appropriateness of the model for such a development was then suggested. This could be achieved by comparing the input-process-output of the model (i.e. 'superstructure' section) with some real project data. Thus, the planner made arrangements to obtain some real data from a recently completed reinforced concrete multi-

storey office building project. A new interview date was arranged to get the information from the planner.

Interview No 11 :

At: Wimpey Construction, Bristol.

Dates: 8/11/95

Aim: Assessment of Information Obtained From a 'Reinforced Concrete Frame Detailed Contract Programme' and Related Data.

During this interview the planner provided a detailed programme for the reinforced concrete frame of a multi-storey office development and related data for the assessment of future developments of the integrated computer model.

The project was a multi-storey (6 floors) reinforced concrete office buildings. The storeys of the building had little differences in their architectural and structural plans. Thus the detailed contract programme showed the same duration for formwork/reinforcement/concrete for every floor. This approach was exactly the same in the Simulation Model where small variances of the floors were ignored and duration values for each floor were calculated as an average. However, as the programme was a contract programme, casting, curing and striking of concrete were presented in much more detail than in the Simulation Model developed by the author.

In addition to the contract programme, a priced bill of quantities for the concrete frame was provided. However, this only included the labour unit costs in £/hr values not in £/unit

of quantity values. This approach is totally different from the unit cost estimating approach undertaken within the model. Thus, it was not possible to use these unit costs within the model.

APPENDIX 3 OUTPUT RATE CALCULATION GUIDE FROM MANUFACTURERS JCB AND WACKER

Output Performance

The basic cycle time - BCT, is made up of various operational timings as the following tables show.

LOADER

		3CX	2CX	
Raise to full height from ground level	=	3.3	6.0	secs
Lower from full height to ground level	=	2.2	4.1	secs
Dump at full height	=	1.6	1.8	secs
Crowd at full height	=	1.3	2.2	secs

EXCAVATOR

	2CX	3CX 3C	3D	4CT	
Boom raise to full height from ground level	4.0	2.4	2.4	2.4	secs
Boom lower from full height to ground level	3.2	2.7	2.7	2.8	secs
Dipper in	3.1	3.4	3.6	3.8	secs
Dipper out	2.5	2.2	2.5	2.8	secs
Bucket crowd (speed position)	3.2	2.7	2.1	2.2	secs
Bucket dump (speed position)	2.5	1.7	1.7	1.8	secs
Slew 180° (left and right)	3.8	4.0	4.3	4.4	secs
Stabiliser in	5.5	2.8	2.8	5.0	secs
Stabiliser out	4.0	2.6	2.6	5.5	secs

For calculation purposes BCT is as follows:-

3C = 8 secs, 3CX = 8 secs, 3D = 8 secs,

4CT = 9 secs, 2CX = 10.5 secs.

Output performance is governed by many variable factors and at best can only be estimated to give a guide.

EXCAVATING

To calculate output per hour four main factors affect the performance, dig depth, material, target, and slew angle. One other factor can alter the output and that is the operator, but for these calculations this is taken to be of average status.

Compensation figures to be added to BCT to give corrected cycle time

Dig Depth	Factor
0-1 m	1
1-2 m	2
2-3 m	4
3-4 m	6
4-5 m	8
5-6 m	10

Target Type	Factor
Open Dump	0
Small Target	5
Target with obstruction	7

Slew Angle	Factor
0°-30°	0
30°-60°	1
60°-90°	2

Dig Material	Factor
Soft Earth - Loose Material - Stock Pile	0
Medium Dig - Clay	2
Medium Dig - Packed Earth - Soil/25% Rock	4
Medium/Hard Dig - Hard Soil/50% Rock	10
Tough Dig - Sandstone,Shale,Limestone,Frost	16

APPENDIX 4 CONSTRAINT INEQUALITIES FOR A REINFORCED CONCRETE MULTI STOREY BUILDING

Duration Values:

Pr : Project duration under normal conditions.

X(i) : Starting point (in days) of activity (i).

dur(i) : Duration of activity (i) under normal conditions.

durm(i): Duration of activity (i) under crash conditions.

(i)	Activity
Xp	End of the project
(1)	Excavate topsoil.
(2)	Excavate reduce levels.
(3)	Excavate basements.
(4)	Excavate trenches.
(5)	Compact bottoms of excavations.
(6)	Blinding concrete.
(7)	Formwork for foundations.
(8)	Reinforcement for foundations.
(9)	Concrete for foundations.
(10)	Roof asphalt.
(11)	Internal brickwork.
(12)	Internal blockwork.
(13)	External brickwork.
(14)	External blockwork.
(15)	Internal finishes.
(16)	Wall finishes.
(17)	Floor finishes.
(18)	Ceiling finishes.
(19)	External finishes.
(20)	Lift Services.
(21)	Mechanical Services.
(22)	Electrical Services.
(23)	Formwork for slab/beams.
(24)	Reinforcement for slab/beams.
(25)	Concrete for slab/beams.
(26)	Formwork for columns/walls.
(27)	Reinforcement for columns/walls.
(28)	Concrete for columns/walls.
(29)	Formwork for staircases.
(30)	Reinforcement for staircases.
(31)	Concrete for staircases

Table 1 : Numbers Used for the Activities While Formalising the Constraints.

Lag Values

Lag	Activity Numbers:	
(1)	(1) & (2)	
(2)	(2) & (3), (2) & (4), (2) & (5)	
(5)	(5) & (6)	
(6)	(6) & (7)	
(7)	(7) & (8)	
(8)	(8) & (9)	
(9)	(9) & (23)	
(10)	(23) & (27)	
(11)	(27) & (23)*	*1st floor
(12)	(23)* & (29)	
(13)	(29)** & (11), (29)** & (12)	**Floor number
(14)	(23)** & (13), (23)** & (14)	changeable from
(15)	(27) & (26)	ground floor to
(16)	(26) & (28)	the roof (see the
(17)	(23) & (24)	constraints below)
(18)	(24) & (25)	
(19)	(29) & (30)	
(20)	(30) & (31)	
(21)	(13) & (15), (14) & (15)	
(22)	(13) & (16), (14) & (16)	
(23)	(13) & (17), (14) & (17)	
(24)	(13) & (18), (14) & (18)	
(25)	(13) & (19), (14) & (19)	
(26)	(13) & (20), (14) & (20)	
(27)	(13) & (21), (14) & (21)	
(28)	(13) & (22), (14) & (22)	
(29)	(25)*** & (10)	***Roof

Table 2 Numbers Used for the Lag Values While Formalising the Constraints.

Constraints:

nofloor = number of floors

aa = 6*nofloor-5

bb = 2*aa

To enable the below formulations and also the coding of the programme OPT2 more clearly, it should be noted that the coefficients of the formulations have been stored in the data files by starting with the activities in the same order in Table 1 and then storing the values for formwork, reinforcement, concrete for slabs/beams and columns/walls for the remaining floors which have not been included in Table 1. The coefficients of the T(i) values are then stored. Finally the coefficients of the X(i) and T(i) values for staircases are stored. The same order is employed in the programme OPT1 while storing the coefficients of variables within the objective function, i.e., U(i) values. It should finally be noted here that the lag values are the ones input for the acceleration of the project.

'excavations and foundations'

$X(2)-X(1) + T(1) \geq lag(1)$

$T(1) \leq dur(1)-durm(1)$

$X(3)-X(2) + T(2) \geq lag(2)$

$T(2) \leq dur(2)-durm(2)$

$X(4)-X(2) + T(2) \geq lag(2)$

$X(5)-X(2) + T(2) \geq lag(2)$

$X(6)-X(5) + T(5) \geq lag(5)$

$T(5) \leq dur(5)-durm(5)$

$X(6)-X(3) + T(3) \geq lag(2)$

$T(3) \leq dur(3)-durm(3)$

$X(6)-X(4) + T(4) \geq lag(2)$

```
T(4) ≤ dur(4)-durm(4)

X(7)-X(6) + T(6) ≥ lag(6)

T(6) ≤ dur(6)-durm(6)

X(8)-X(7) + T(7) ≥ lag(7)

T(7) ≤ dur(7)-durm(7)

X(9)-X(8) + T(8) ≥ lag(8)

T(8) ≤ dur(8)-durm(8)

' ground floor slabs/beams'

X(23)-X(9) + T(9) ≥ lag(9)

T(9) ≤ dur(9)-durm(9)

X(24)-X(23) + T(23) ≥ lag(17)

T(23) ≤ dur(23)-durm(23)

X(25)-X(24) + T(24) ≥ lag(18)

T(24) ≤ dur(24)-durm(24)

'ground to first floor columns/walls'

X(27)-X(23) + T(23) ≥ lag(10)

X(26)-X(27) + T(27) ≥ lag(15)

T(27) ≤ dur(27)-durm(27)

X(28)-X(26) + T(26) ≥ lag(16)

T(26) ≤ dur(26)-durm(26)

'other floors slabs/beams & columns/walls'

For i= 1 to nofloor-2

X(6i+32)-X(6i+30)+T(6i+30) ≥ lag(11)

T(6i+30) ≤ dur(27)-durm(27)

next
```

For i= 1 to nofloor-1

X(6(i-1)+36)-X(6(i-1)+32)+T(6(i-1)+32) ≥ lag(10)

T(6(i-1)+32) ≤ dur(23)-durm(23)

next

'roof slab'

X(10)-X(33+aa) + T(33+aa) ≥ lag(29)

T(33+aa) ≤ dur(25)-durm(25)

'ground to first floor staircases'

X(29)-X(23) + T(23) ≥ lag(12)

X(30)-X(29) + T(29) ≥ lag(19)

T(29) ≤ dur(29)-durm(29)

X(31)-X(30) + T(30) ≥ lag(20)

T(30) ≤ dur(30)-durm(30)

'other floor staircases'

for i=2 to number of floors -2

X(63+bb+3(i-1))-X(62+bb+3*(i-1)30) + T(62+bb+3(i-1)) ≥ dur(31)

T(62+bb+3(i-1)) ≤ dur(31)-durm(31)

'internal cladding'

X(11)-X(29) + T(29) ≥ lag(13)

X(12)-X(29) + T(29) ≥ lag(13)

'external cladding'

X(13)-X(23) + T(23) ≥ lag(14)

X(14)-X(23) + T(23) ≥ lag(14)

'finishes & services'

X(15)-X(13) + T(13) ≥ lag(21)

T(13) ≤ dur(13)-durm(13)

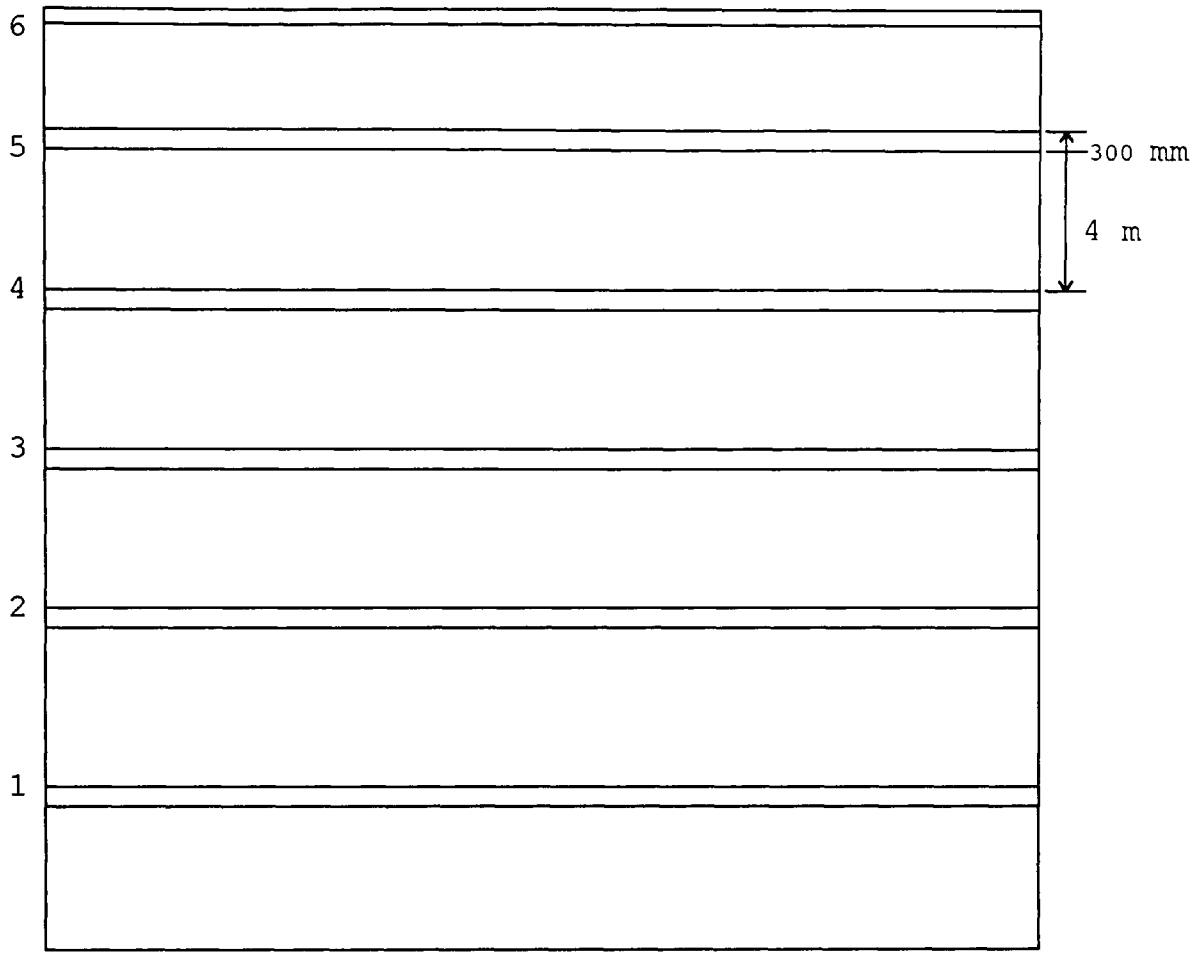
```
X(15)-X(14) + T(14) ≥ lag(21)

T(14) ≤ dur(14)-durm(14)

X(16)-X(13) + T(13) ≥ lag(22)

X(16)-X(14) + T(14) ≥ lag(22)

X(17)-X(13) + T(13) ≥ lag(23)

X(17)-X(14) + T(14) ≥ lag(23)

X(18)-X(13) + T(13) ≥ lag(24)

X(18)-X(14) + T(14) ≥ lag(24)

X(19)-X(13) + T(13) ≥ lag(25)

X(19)-X(14) + T(14) ≥ lag(25)

X(20)-X(13) + T(13) ≥ lag(26)

X(20)-X(14) + T(14) ≥ lag(26)

X(21)-X(13) + T(13) ≥ lag(27)

X(21)-X(14) + T(14) ≥ lag(27)

X(22)-X(13) + T(13) ≥ lag(28)

X(22)-X(14) + T(14) ≥ lag(28)

' the end of the project'

Xp -X(10) + T(10) ≥ durm(10)

T(10) ≤ dur(10)-durm(10)

Xp - X(15) + T(15) ≥ durm(15)

Xp - X(16) + T(16) ≥ durm(16)

Xp - X(17) + T(17) ≥ durm(17)

Xp - X(18) + T(18) ≥ durm(18)

Xp - X(19) + T(19) ≥ durm(19)

Xp - X(20) + T(20) ≥ durm(20)
```

$$Xp - X(21) + T(21) \geq durm(21)$$

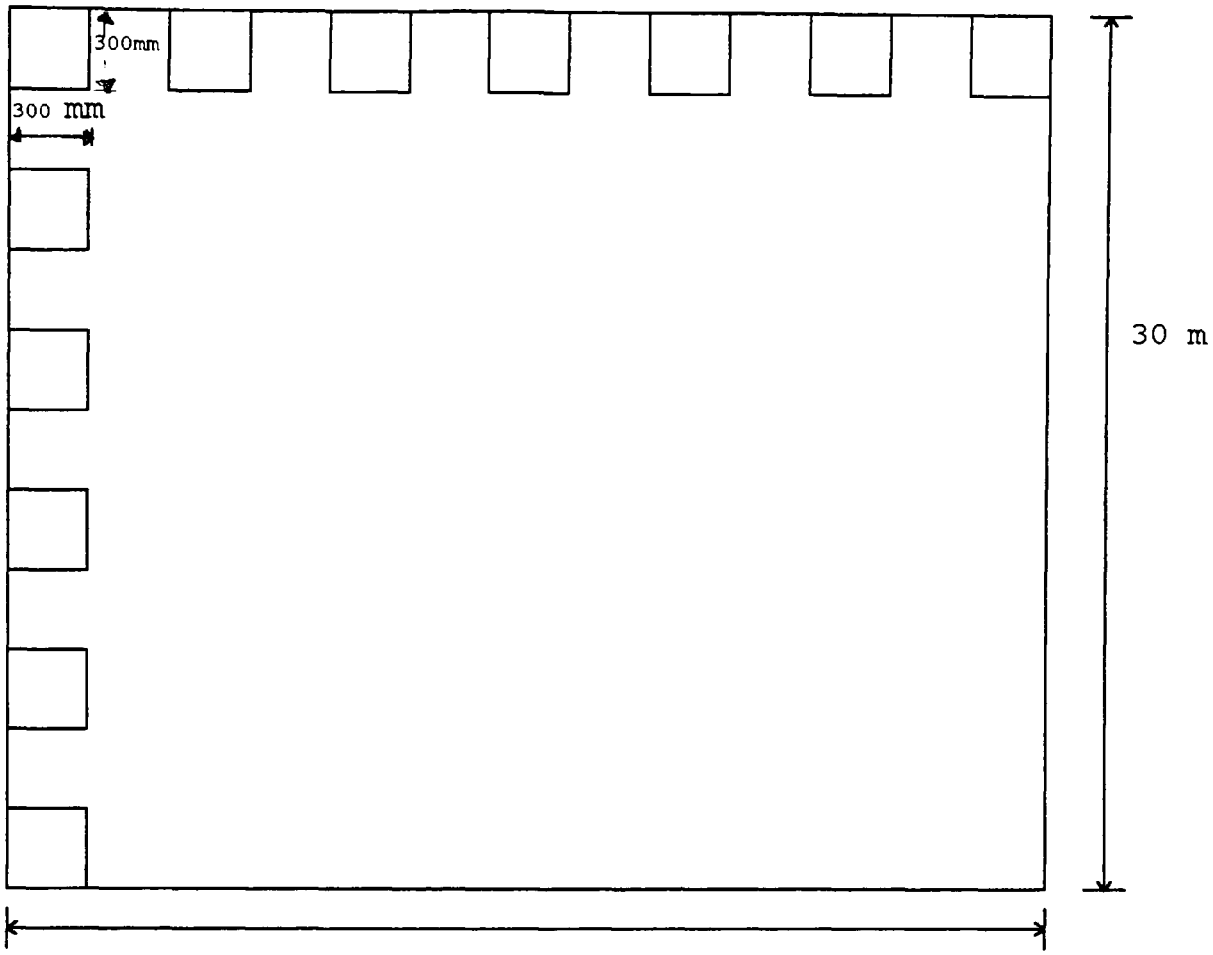
$$Xp - X(22) + T(22) \geq durm(22)$$

$$Xp \leq Pr$$

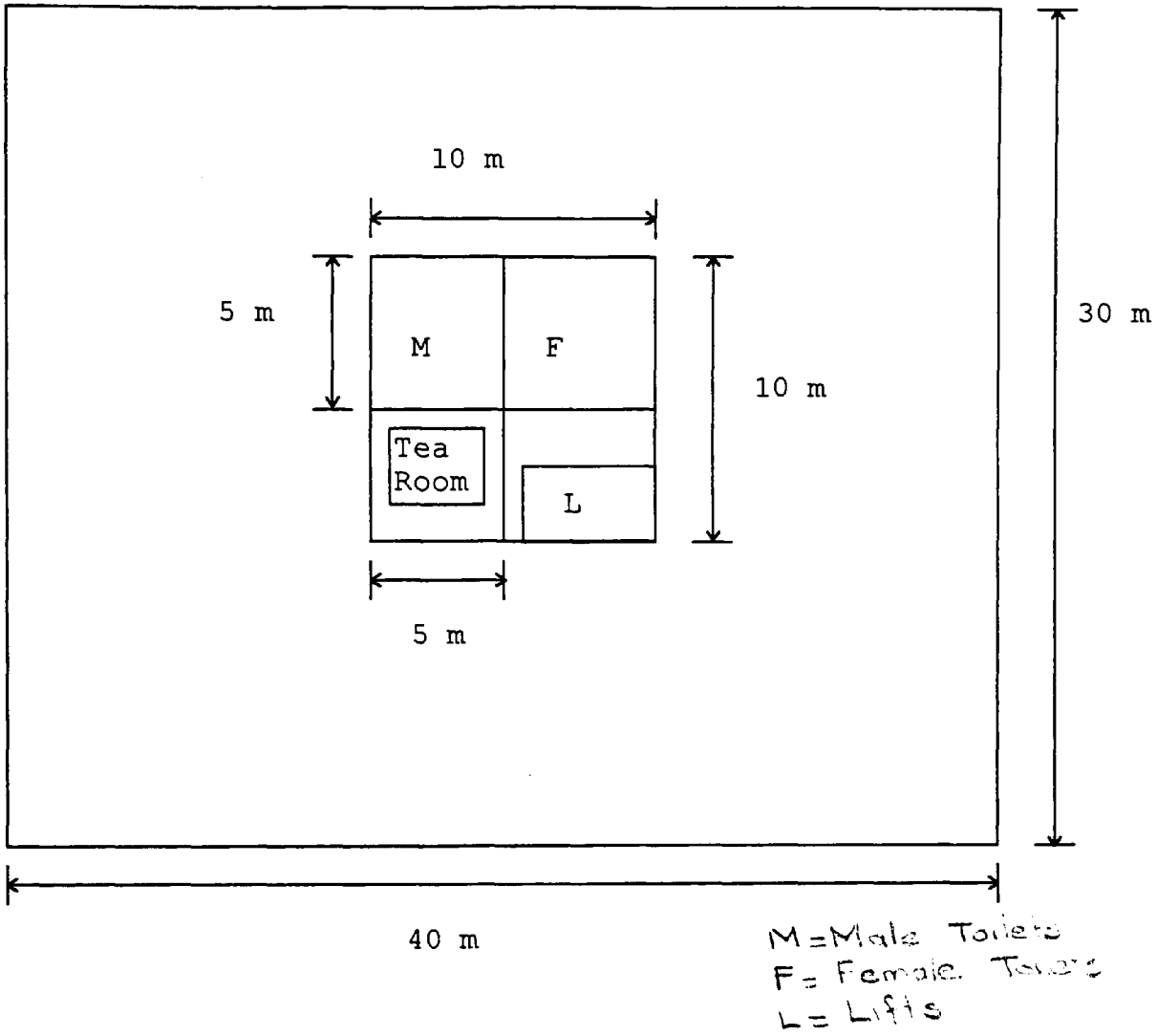
APPENDIX 5 THE DRAWINGS AND THE BILL OF QUANTITIES FOR THE HYPOTHETICAL BUILDING



ELEVATION (not to scale)

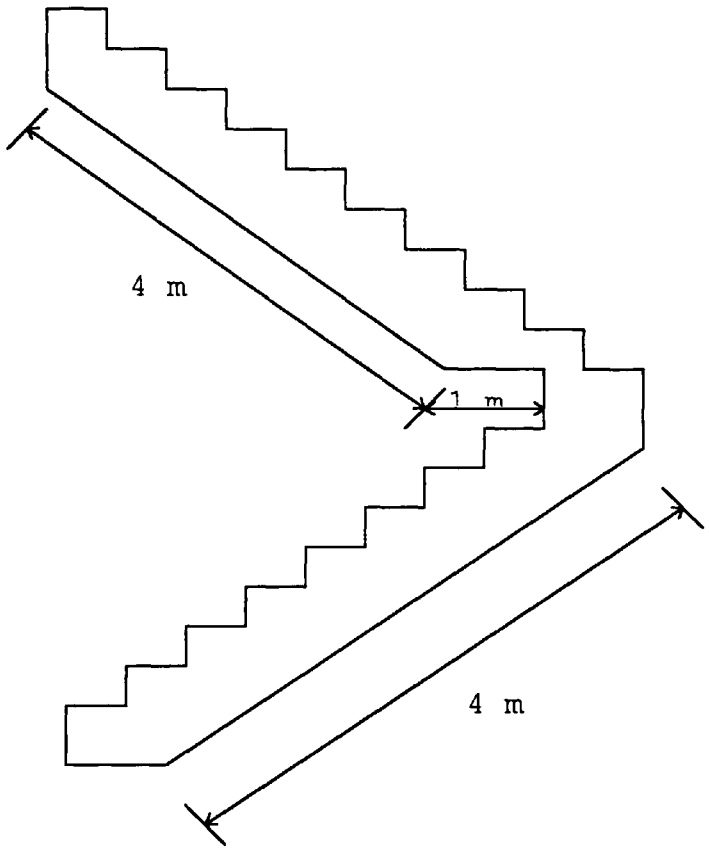


PLAN SHOWING THE COLUMNS

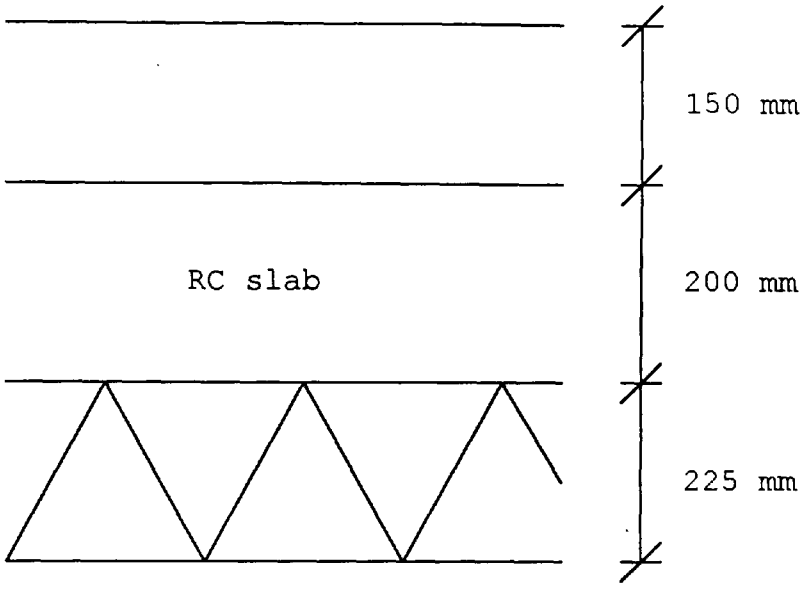


7 x 6 = 42 columns

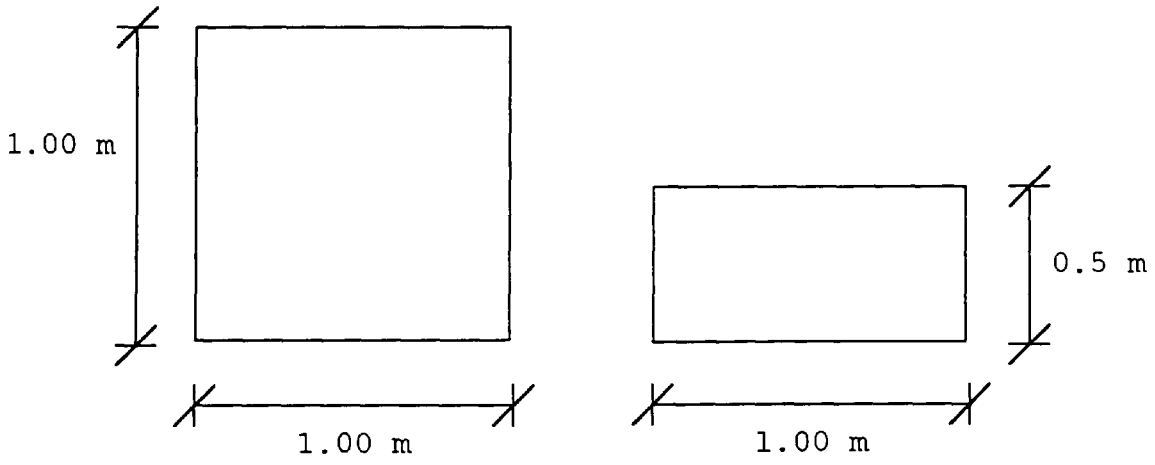
INTERNAL PARTITIONS (100 mm) (not to scale)
(All the floors are identical)



STAIRCASES (not to scale)



GROUND FLOOR SLAB (TYPICAL) (not to scale)



BASE (2 No.) (not to scale)

	Substructure Excavation and Earthwork.					
	Site Preparation					
	Note: The contractor should note from the information contained in the Trial Fit Lags that clay will be encountered below the surface of the ground during excavations.					
a	Excavate over site to remove top soil and vegetation not to be preserved.					
	average 150 mm deep	180	m3			
b	Excavate topsoil to be preserved					
	average 150 mm deep	90	m3			
	Excavation					
c	Excavate to reduce level					
	max. depth not exceeding 0.99 m	510	m3			
d	Excavate trenches to receive foundations					
	max. depth not exceeding 0.99 m	135	m3			
	Concrete work					
	In-situ Concrete					
	Total volume of concrete in this Element - 385 m3					
	Design mix C10 mass concrete as described .					
e	Bed as blinding layer, laid on earth, not exceeding 100 mm thick	9	m3			

	Design mix C20 mass concrete as described.						
a	Isolated foundation base to columns and piers (in 2 no.)						
	exceeding 300 mm thick	1	m3				
	Design mix C35 reinforced concrete as described.						
b	Bed						
	not exceeding 300 mm thick	240	m3				
c	Foundation in trench						
	exceeding 300 mm thick	135	m3				
	Reinforcement						
	High yield steel bar reinforcement as described.						
d	Bars, straight or bent, in foundations and walls						
	6 mm diameter						
e	Ditto,						
	8 mm diameter						
f	Ditto,						
	10 mm diameter						
g	Ditto,						
	12 mm diameter	10125	kg				
h	Ditto,						
	16 mm diameter						
i	Ditto,						
	20 mm diameter						
j	Ditto,						
	25 mm diameter						

	Fabric Reinforcement as described					
a	Reinforcement in beds and ground beams					
	A98 - A393 at 200 mm side and end laps	1200	m2			
	Formwork					
	Formwork to provide ordinary finish to non-exposed surface (Type A) as described:					
b	Vertical face of foundation or edge of bed, not exceeding 250 mm high	140	m			

	Reinforced Concrete Structure					
	Columns and Walls					
	Concrete work					
	In-situ concrete					
a	Total in-situ concrete in this Sub Element - 90 m3					
	Design Mix C35 reinforced concrete as described .					
b	Isolated column,					
	0.03 - 0.1 m2 sectional area	90	m3			
	Reinforcement					
	High yield steel bar reinforcement as described.					
c	Bars, straight or bent, in walls					
	6 mm diameter					
d	Ditto,					
	8 mm diameter					
e	Ditto,					
	10 mm diameter					
f	Ditto,					
	12 mm diameter					
g	Ditto,	9000	kg			
	16 mm diameter					
h	Ditto,					
	20 mm diameter					
i	Ditto,					
	25 mm diameter					

	Formwork					
	Formwork to provide an ordinary finish to non-exposed surfaces (Type A) as described:					
a	Isolated rectangular column,					
	200 - 400 mm width (in 252 no.)	302	m2			
	Upper Floors and Associated Beams					
	Concrete Work					
	In concrete framed structure					
	In-situ concrete					
	Total in-situ concrete in this Sub-Element - 2160 m3					
	Design mix C35 reinforced concrete as described					
b	Suspended slab,					
	150-300 mm thick	2160	m3			
	Reinforcement					
	High yield steel bar reinforcement as described.					
c	Bars, straight or bent, in suspended slabs					
	6 mm diameter					
d	Ditto,					
	8 mm diameter					
e	Ditto,					
	10 mm diameter	172800	kg			
f	Ditto,					
	12 mm diameter					

	High yield steel bar reinforcement as described.					
a	Bars, straight or bent, in suspended slabs 16 mm diameter					
b	Ditto, 20 mm diameter					
c	Ditto, 25 mm diameter					
	Formwork					
	Formwork to provide ordinary finish to non-exposed surfaces (Type A) as described:					
d	Soffit to suspended slab, slab 200-300 mm thick, height exceeding 3.5m, (in 6 no. separate surfaces)	720	m2			
	Staircases					
	Concrete Work					
	In concrete Framed structure					
	In-situ concrete					
	Total in-situ concrete in this Sub Element - 18 m3.					
	Design mix C35 reinforced concrete as described.					
e	Steps and stairs	18	m3			
	Reinforcement					
	High yield steel bar reinforcement as described.					
f	Bars, straight or bent, in staircases and landings, 6 mm diameter					

	High yield steel bar reinforcement as described.					
a	Bars, straight or bent, in staircases and landings,					
	8 mm diameter					
b	Ditto,					
	10 mm diameter					
c	Ditto,					
	12 mm diameter	1350	kg			
d	Ditto,					
	16 mm diameter					
	Formwork					
	Formwork to provide an ordinary finish to non-exposed surfaces (Type A) as described:					
e	Soffits of staircases,					
	not exceeding 200 mm thick, (in 6 no. separate surfaces)	54	m2			

	Roofing					
a	Asphalt	1200	m2			

	Brickwork and Blockwork					
	External Walls					
	Brickwork					
a	Common bricks, BS 3921, PC £80 per 1000 in cement mortar (1:3)					
	Half brick wall	2520	m2			
	Blockwork					
	Precast concrete blocks, (BS 6073), compressive strength 7 N/mm2, in gauged mortar					
b	150 mm thick	2520	m2			
	Internal Walls					
	Blockwork					
c	Precast concrete blocks, (BS 6073), compressive strength 7 N/mm2, in gauged mortar					
	100 mm thick	1332	m2			

APPENDIX 6 THE INPUT SCREENS FROM THE SIMULATION MODEL USED FOR THE INPUT OF QUANTITIES OF MATERIALS FOR THE HYPOTHETICAL BUILDING

INPUT VALUES WHERE REQUIRED.

	AREA m2	DEPTH m	VOLUME m3
(1)Excavated topsoil to be removed	90.0	2.0	180.0
(2)Excavated topsoil to be preserved	90.0	1.0	90.0

Select the most appropriate definition of your excavated material from 1 to 5.

(1) Soft earth or loose material
(2) Clay
(3) Medium packed earth or soil that contains 25% rock
(4) Medium to hard packed earth or soil that contains 50% rock
(5) Sandstone, shale, limestone, frost

[3]

Please specify the total volume (m3) of excavations for the following categories of depth (m).

	0-0.99	1-1.99	2-2.99	3-3.99	4-4.99	5-5.99	6+
Excavate to reduce level	510.0	.0	.0	.0	.0	.0	.0
Excavate for basements	.0	.0	.0	.0	.0	.0	.0
Excavate for trenches	135.0	.0	.0	.0	.0	.0	.0
Type of material at the above depths Please specify 0 to 5.	3	3	3	3	3	3	3

(0) If there is no excavation at that category of depth.
(1) Soft earth or loose material
(2) Clay
(3) Medium earth or soil with 25% rock
(4) Medium to hard packed earth/soil with 50% rock
(5) Sandstone, shale, limestone, frost

	Backhoe/Crawler	Bucket Capacity m3	Basic Cycle time secs
(1)	JCB 3C, 3CX, 3D	0.30	8.0
(2)	JCB 4CT	0.17	9.0
(3)	JCB 2CX	0.24	10.5
(4)	JCB 801	0.04	15.0
(5)	JCB 803	0.11	15.0
(6)	JS 110	0.45	14.0
(7)	JS 150	0.70	14.0
(8)	JS 240	1.25	14.0
(9)	JS 300	1.75	14.0
(10)	JS 450	2.25	14.0

Specify a plant number (1 to 10), and the amount to be used on each activity.

	Plant	Amount	Plant	Amount	Volume of excavation
Excavate topsoil	1	1	6	2	270.0 m3
Excavate to reduce level	1	1	6	2	510.0 m3
Excavate for basements	0	1	6	2	.0 m3
Excavate for trenches	1	1	6	2	135.0 m3

What method will be used to remove excavated material from the excavation area?

(1) Excavator will dump material in an open area.
(2) Excavator will dump material into a vehicle.
(3) Excavator will dump material into a vehicle
but there are site obstructions.

3

PLAIN DENSE AGGREGATE CONCRETE FOUNDATIONS

Enter the total volume (m3) of plain concrete for the following items.

	Thickness (mm) <100	100-150	150-300	300+	
Bed as blinding layer	9.0	.0	.0	.0	m3
Fdns in trenches	.0	.0	.0	135.0	m3

	Cross sectional area (m2) <0.03	0.03-0.1	0.1-0.25	0.25+	
Ground beams	.0	.0	.0	.0	m3

Isolated foundation bases to columns and piers 1.0 m3

REINFORCED CONCRETE FOUNDATIONS

Enter the total volume (m3) of reinforced concrete for the following items.

	Thickness (mm) <100	100-150	150-300	300+	
Beds	.0	.0	240.0	.0	m3
Fdns in trenches	.0	.0	.0	.0	m3
Walls	.0	.0	.0	.0	m3

	Cross sectional area (m2) <0.03	0.03-0.1	0.10-0.25	0.25+	
Ground beams	.0	.0	.0	.0	m3
Isolated columns	.0	.0	.0	.0	m3

Isolated foundation bases to columns and piers .0 m3

REINFORCEMENT TO CONCRETE FOUNDATIONS

Please specify weights (ton) for the following categories of reinforcement.

STEEL BARS (BS 4449) (Mild & High yield steel bars)

	Straight/ bent bars							Links, stirrups, binders			
Dia (mm)	6	8	10	12	16	20	25+	6	8	10	
	.0	.0	.0	.0	.0	.0	10.1	.0	.0	.0	(ton)

Please specify areas (m2) for the following categories of reinforcement.

STEEL FABRIC REINFORCEMENT (BS 4483)

Fabric Reinforcement to:	Ground slabs/beds	Walls	
A98 - A393 at 200mm side & end laps	1200.0	.0	m2
B196 - B1131 at 100mm side & 200mm end laps	.0	.0	m2
C283 - C785 at 100mm side & 400mm end laps	.0	.0	m2

FORMWORK TO CONCRETE/REINFORCED CONCRETE FOUNDATIONS

Enter the total area (m2)/lenght (m) of formwork for the following categories.

Height (m)	Vertical face of Fdns/Beds.		Ground beams.	
>1.00	.0	m2	.0	m2
<0.25	140.0	m	.0	m
0.25-0.50	.0	m	.0	m
0.50-1.00	.0	m	.0	m

Vertical faces of walls (one or both sides , 1 side measured) .0 m2

Enter the total height (m)/area (m2) of formwork to the sides of isolated columns for the following categories of width (mm).

Width (mm)	<200	200 - 400	400 - 600	Width (mm)	600+
Height (m)	.0	.0	.0	Area (m2)	.0

Please enter the number of floors of the building starting after foundations until the roof.

6

FORMWORK FOR COLUMNS AND WALLS

Please input the length (m) of formwork for all isolated rectangular columns.

Width (mm)	Length (m)
< 200	.0
200 -400	302.0
400 -600	.0
> 600	.0

Please input the area (m2) of formwork for the sides of isolated circular columns.

Dia. (mm)	Area (m2)
< 300	.0
300 -600	.0
600 -900	.0

Please input the area (m2) of formwork for vertical face walls.

One side of wall shuttered	.0	m2
Both sides of wall shuttered	.0	m2

REINFORCEMENT FOR COLUMNS AND WALLS

Please specify weights (ton) for the following categories of reinforcement.

STEEL BARS (BS 4449) (Mild & High yield steel bars)

	Straight/ bent bars							Links, stirrups, binders			
Dia (mm)	6	8	10	12	16	20	25	6	8	10	
COLUMNS .	0	.0	.0	.0	.0	.0	9.0	.0	.0	.0	(ton)
WALLS	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	(ton)

REINFORCED CONCRETE TO WALLS AND COLUMNS

Please input the volume (m3) of reinforced concrete to walls.

Thickness (mm)	Aggregate Light	Dense
< 100	.0	.0
100 -150	.0	.0
150 -300	.0	.0
> 300	.0	.0

Please input the volume (m3) of reinforced concrete to isolated columns.

C/S Area (m2)	Aggregate Light	Dense
< 0.03	.0	.0
0.03 -0.10	90.0	.0
0.10 -0.25	.0	.0
> 0.25	.0	.0

FORMWORK AND REINFORCED CONCRETE FOR STAIRCASES

Please input the area (m2) of formwork for soffits of staircases not exceeding 200 mm thickness.

Height (m)	Area (m2)
< 3.5	54.0

Please input the area (m2) of formwork for soffits of landings not exceeding 200 mm thickness.

Height (m)	Area (m2)
< 3.5	.0

Please input the volume (m3) of reinforced concrete for steps and staircases with 25 kN/mm2 concrete.

	Aggregate	
	Light	Dense
Volume (m3)	18.0	.0

REINFORCEMENT FOR STAIRCASES

Please specify weights (ton) for the following categories of reinfocement. for staircases.

STEEL BARS (BS 4449) (Mild & High yield steel bars)

	Straight/ bent bars							Links, stirrups, binders			
Dia (mm)	6	8	10	12	16	20	25	6	8	10	
	.0	.0	.0	.0	.0	.0	1.4	.0	.0	.0	(ton)

FORMWORK FOR SLABS

Please input the area (m2) for formwork to soffits of slabs.

Thickness (mm)	Height (m)	
	< 3.5	> 3.5
< 200	.0	.0
200 -300	.0	7200.0
300 -400	.0	.0

Please input the lenght (m) for the formwork to the vertical edges of slabs.

Depth (mm)	Length (m)
< 250	.0
250 -500	.0

REINFORCEMENT FOR SLABS AND BEAMS

Please specify weights (ton) for the following categories of reinfocement.

STEEL BARS (BS 4449) (Mild & High yield steel bars)

	Straight/ bent bars							Links, stirrups, binders			
Dia (mm)	6	8	10	12	16	20	25	6	8	10	
SLABS .	0	.0	.0	.0	.0	.0	172.8	.0	.0	.0	(ton)
BEAMS	.0	.0	.0	.0	.0	.0	.0	.0	.0	.0	(ton)

REINFORCED CONCRETE FOR SLABS AND BEAMS

Please input the volume (m3) of reinforced concrete for suspended slabs, isolated beams and deep beams.

SUSPENDED SLABS

Thickness (mm)			
< 100	100 -150	150 -300	> 300
.0	.0	2160.0	.0

ISOLATED BEAMS

Cross Sectional Area (m2)			
< 0.03	0.03 -0.10	0.10 -0.25	> 0.25
.0	.0	.0	.0

DEEP BEAMS

Cross Sectional Area (m2)		
0.03 -0.10	0.10 -0.25	> 0.25
.0	.0	.0

BLOCKWORK

Please input the area (m2) of the blockwork for the following.
Precast concrete blocks,(BS 6073), compressive strength 7 N/mm2,
in gauged mortar.

	Walls and partitions					Skins of hollow blocks			
	Solid blocks		Hollow blocks			Solid blocks		Hollow blocks	
Thickness (mm)		*		*	Thickness (mm)		*		*
60	.0	.0	.0	.0	90	.0	.0	.0	.0
75	.0	.0	.0	.0	115	.0	.0	.0	.0
100	.0	.0	.0	.0	125	.0	.0	.0	.0
150	2520.0	.0	.0	.0	140	.0	.0	.0	.0
200	.0	.0	.0	.0	190	.0	.0	.0	.0
215	.0	.0							

	Piers			
	Solid blocks		Hollow blocks	
Thickness (mm)		*		*
100	.0	.0	.0	.0
150	.0	.0	.0	.0
190	.0	.0	.0	.0
200	.0	.0	.0	.0
215	.0	.0	.0	.0

BRICKWORK

Please input the area (m2) of the brickwork for the following.
Facing Bricks PC £160 per 1000 in gauged mortar (1:2:9);flush pointing
as the work proceeds.

	Thickness (mm)	
	102.5	215
Area (m2) of:-		
- Walls	.0	2520.0
- Skins of hollow walls	.0	

BLOCKWORK

Please input the area (m2) of the blockwork for the following.
Precast concrete blocks,(BS 6073), compressive strength 7 N/mm2,
in gauged mortar.

Walls and partitions					Skins of hollow blocks				
	Solid blocks		Hollow blocks			Solid blocks		Hollow blocks	
Thickness (mm)	*		*		Thickness (mm)	*		*	
60	.0	.0			90	.0	.0		
75	.0	.0			115	.0	.0	.0	.0
100	1332.0	.0	.0	.0	125	.0	.0	.0	.0
150	.0	.0	.0	.0	140	.0	.0	.0	.0
200	.0	.0	.0	.0	190	.0	.0	.0	.0
215			.0	.0					

Piers				
	Solid blocks		Hollow blocks	
Thickness (mm)	*		*	
100	.0	.0	.0	.0
150	.0	.0	.0	.0
190	.0	.0	.0	.0
200	.0	.0	.0	.0
215	.0	.0	.0	.0

* Fair face and flush pointing (one side or both sides)

Please input the quanties and unit costs for Mastic Asphalt BS 988.
20 mm Two coat work rubbed with fine sand; to concrete base: flat coverings:

	Quantity	ulc (£/hr)	umc(£/m2)
over 300 mm wide	.0	.0	.0
not exceeding 150 mm wide	1200.0	3.5	3.5
150 - 300 mm wide	.0	.0	.0

APPENDIX 7 THE INPUT SCREENS FOR THE VALUES OF GANG SIZES, OUTPUT RATES AND LAG BETWEEN TASKS ACCORDING TO THE SECOND PLANNER'S DATA AND CORRESPONDING OUTPUT SCREENS.

	Backhoe/Crawler	Bucket Capacity m3	Basic Cycle time secs
(1)	JCB 3C, 3CX, 3D	0.30	8.0
(2)	JCB 4CT	0.17	9.0
(3)	JCB 2CX	0.24	10.5
(4)	JCB 801	0.04	15.0
(5)	JCB 803	0.11	15.0
(6)	JS 110	0.45	14.0
(7)	JS 150	0.70	14.0
(8)	JS 240	1.25	14.0
(9)	JS 300	1.75	14.0
(10)	JS 450	2.25	14.0

Specify a plant number (1 to 10), and the amount to be used on each activity.

	Plant	Amount	Plant	Amount	Volume of excavation
Excavate topsoil	1	1	6	2	270.0 m3
Excavate to reduce level	1	1	6	2	510.0 m3
Excavate for basements	0	1	6	2	.0 m3
Excavate for trenches	1	1	6	2	135.0 m3

What method will be used to remove excavated material from the excavation area?

(1) Excavator will dump material in an open area.
(2) Excavator will dump material into a vehicle.
(3) Excavator will dump material into a vehicle
but there are site obstructions.

3

GANG SIZE TO BE EMPLOYED

Please state the normal and maximum gang sizes that would be employed to undertake the following activities for foundations.
(If overtime work will be employed to accelerate work, input "0" as maximum gang sizes)

	Normal	Maximum
FORMWORK	2	5
REINFORCEMENT	2	4
CONCRETE	2	7
BLINDING LAYER	2	4

Please input the output rates if required.

	Output rate	
Blinding concrete	4.0	hr/m3
Concrete for: isolated bases	.4	hr/m3
beds	.3	hr/m3
trenches	.4	hr/m3
ground beams	.0	hr/m3
walls	.0	hr/m3
isolated columns	.0	hr/m3
Reinforcement: re-bars	19.0	hr/ton
mesh	.2	hr/m2
Formwork	.3	hr/m2 (m)

GANG SIZE TO BE EMPLOYED

Please state the normal and maximum gang sizes to be employed to undertake the following activities.

FORMWORK FOR:-	Normal	Maximum
- Columns and Walls	4	8
- Staircases	2	4
- Slabs and Beams	6	12

REINFORCEMENT FOR:-	Normal	Maximum
- Columns and Walls	2	6
- Staircases	1	2
- Slabs and Beams	6	9

REINFORCED CONCRETE FOR:-	Normal	Maximum
- Columns and Walls	2	4
- Staircases	2	3
- Slabs and Beams	4	8

Please input the required output rate values.

FORMWORK FOR:-	Output rate (hr/m2)
- Columns and Walls	1.1
- Staircases	.5
- Slabs and Beams	.5

REINFORCEMENT FOR:-	Output rate (hr/ton)
- Columns and Walls	22.0
- Staircases	25.0
- Slabs and Beams	20.0

REINFORCED CONCRETE FOR:-	Output rate (hr/m3)
- Columns and Walls	2.5
- Staircases	.6
- Slabs and Beams	.3

Please state the normal and maximum gang sizes that would be employed for external blockwork and brickwork.

	Normal	Maximum
Blockwork	8	14
Brickwork	8	14

Please input the related output rates if required.

	Output rate (hr/m2)
Brickwork	1.20
Blockwork	.80

Please state the normal and maximum gang sizes that would be employed for internal blockwork and brickwork.

	Normal	Maximum
Blockwork	3	7
Brickwork	0	0

Please input the related output rates if required.

	Output rate (hr/m2)
Brickwork	0.00
Blockwork	.70

ROOF ASPHALT

	Normal	Maximum
Gang Size	5	8

Please input the output rate of a labour (hr/m2). .8

Excavate Topsoil — Lag values (days) (start to start precedence)

2.0

Excavate to reduce levels

3.0

Excavate & Compact Foundations

1.0

Blinding Foundations

.0

Formwork to Foundations

2.0

Reinforcement to Foundations

3.0

Concrete to Foundations

5.0

Blind & Frc Ground Floor Slab

11.0

Frc Columns/Walls Ground/1st Flr

6.0

1st Floor Frc Slab/Beams

19.0

Frc Staircases

Explanation:
Floor no 1 = Ground FLoor
Floor no 2 = 1st Floor
Floor no 3 = 2nd Floor
No staircases on the last floor

1 th Floor Frc Staircases
3.0
Building Envelope
(Internal Cladding)

3 th Floor Frc Slab/Beam
20.0
Building Envelope
(External Cladding)

Roof Slab/Beam (concrete) 4.0 Roof Asphalt

Please input the lag values (days) between the following activities for reinforced concrete floors (start to start precedence).

	COLUMNS & WALLS		SLABS & BEAMS	STAIRCASES
Reinforcement		Formwork		
	4.0		10.0	2.0
Formwork		Reinforcement		
	8.0		10.0	6.0
Concrete		Concrete		

APPENDIX 8 THE INPUT SCREENS FOR THE UNIT COST VALUES AND INDIRECT COST VALUES FOR THE VALIDATION OF THE MODEL

Please update the following values of plant unit cost (puc) (£/hr).

Bachoe Model	Unit Cost (£/hr)	Crawler Model	Unit Cost (£/hr)
JCB 3C, 3CX, 3D	21.0	JS 110	25.0
JCB 4CT	21.0	JS 150	21.0
JCB 2CX	21.0	JS 240	21.0
JCB 801	21.0	JS 300	21.0
JCB 803	21.0	JS 450	21.0

Vibration Rammer Model (Wacker)	Unit Cost (£/hr)	Vibration Plate Model (Wacker)	Unit Cost (£/hr)
BS 45Y	.0	DPU 2430	.0
BS 60Y/62Y	.0	BPU 2950A	.0
BS 105Y	.0	BPU 3345R	.0
		DPU 6055	.0
		DPU 7060RC	.0

PLAIN DENSE AGGREGATE CONCRETE FOUNDATIONS

Please update the following values of unit labour cost (£/hr) for plain dense aggregate foundations.

	Thickness (mm) <100	100-150	150-300	300+	
Beds	4.8	.0	.0	.0	(£/hr)
Fdns in trenches	.0	.0	.0	4.8	(£/hr)

	Cross sectional area (m2) <0.03	0.03-0.1	0.1-0.25	0.25+	
Ground beams	.0	.0	.0	.0	(£/hr)

Isolated foundation bases to columns and piers .0

Please update the following value of material unit cost (£/m3) for both plain and reinforced concrete grade 25 N/mm2.

70.1 (£/m3)

REINFORCED CONCRETE FOUNDATIONS

Please update the following values of labour unit cost (£/hr) for reinforced concrete grade 25N/mm2 for foundations.

	Thickness (mm) <100	100-150	150-300	300+
Beds	.0	.0	7.1	.0
Fdns in trenches	.0	.0	.0	.0
Walls	.0	.0	.0	.0

	Cross sectional area (m2) <0.03	0.03-0.1	0.10-0.25	0.25+
Ground beams	.0	.0	.0	.0
Isolated columns	.0	.0	.0	.0

Isolated foundation bases to columns and piers .0

REINFORCEMENT TO CONCRETE FOUNDATIONS

Please update the following values of material unit cost (£/kg)

STEEL BARS (BS 4449) (Mild & High yield steel bars)

	Straight/ bent bars							Links, stirrups, binders			
Dia (mm)	6	8	10	12	16	20	25+	6	8	10	
	.0	.0	.0	.0	.0	.0	.4	.0	.0	.0	(£/kg)

Please update the average labour unit cost (£/hr) for steel bars.

6.5 (£/hr)

REINFORCEMENT TO CONCRETE FOUNDATIONS

Please update the following values of material unit cost (muc) (£/m2) & labour material cost (luc) (£/hr).

STEEL FABRIC REINFORCEMENT (BS 4483)

Ground slabs/beds	muc (£/m2)	luc (£/hr)
Fabric Reinforcement to:		
A98- A393 at 200mm side & end laps	1.5	5.6
B196- B1131 at 100mm side & 200mm end laps	.0	.0
C283- C785 at 100mm side & 400mm end laps	.0	.0

Walls	muc (£/m2)	luc (£/hr)
Fabric Reinforcement to:		
A98- A393 at 200mm side & end laps	.0	.0
B196- B1131 at 100mm side & 200mm end laps	.0	.0
C283- C785 at 100mm side & 400mm end laps	.0	.0

FORMWORK TO CONCRETE/REINFORCED CONCRETE FOUNDATIONS

Please update the material unit cost (muc) & labour unit cost (luc) for the following formwork.

Height (m)	Vertical face of Fdns/Beds. muc	luc (£/hr)	Ground beams. muc	luc (£/hr)
>1.00	.0 (£/m2)	.0	.0 (£/m2)	.0
<0.25	7.0 (£/m)	6.1	.0 (£/m)	.0
0.25-0.50	.0 (£/m)	.0	.0 (£/m)	.0
0.50-1.00	.0 (£/m)	.0	.0 (£/m)	.0

	muc	luc (£/hr)
Vertical faces of walls	.0 (£/m2)	.0

Please update the following values of material unit cost (muc) & labour unit cost (luc) for formwork to the sides of isolated columns.

Width (mm)	<200	200 - 400	400 - 600	600+
muc	.0 (£/m)	.0 (£/m)	.0 (£/m)	.0 (£/m2)
luc (£/hr)	.0	.0	.0	.0

FORMWORK FOR COLUMNS AND WALLS

The values for the material unit cost (muc) and the labour unit cost (luc) for the following can be updated if required.

Formwork for all isolated rectangular columns.

Width (mm)	muc (£/m)	luc (£/hr)
<200	.0	.0
200 -400	8.1	12.5
400 -600	.0	.0
>600	.0	.0

Formwork for the sides of isolated circular columns.

Dia. (mm)	muc (£/m2)	luc (£/hr)
< 300	.0	.0
300 -600	.0	.0
600 -900	.0	.0

Formwork for vertical face walls.

	muc (£/m2)	luc (£/hr)
One side of wall shuttered	.0	.0
Both sides of wall shuttered	.0	.0

REINFORCEMENT FOR COLUMNS, WALLS, SLABS, BEAMS AND STAIRCASES

The following are the material unit cost (£/ton) for different categories of reinforcement.

STEEL BARS (BS 4449) (Mild & High yield steel bars)

	Straight/ bent bars							Links, stirrups, binders			
Dia (mm)	6	8	10	12	16	20	25	6	8	10	
	.0	.0	.0	.0	.0	.0	.4	.0	.0	.0	(£/kg)

The following are the average labour unit cost (£/hr) for steel bars.

6.5 (£/hr)

REINFORCED CONCRETE TO WALLS AND COLUMNS

Please update the following values of labour unit cost (£/hr).

Reinforced concrete to walls.

Thickness (mm)	Aggregate Light	Dense	
< 100	.0	.0	(£/hr)
100 -150	.0	.0	(£/hr)
150 -300	.0	.0	(£/hr)
> 300	.0	.0	(£/hr)

Reinforced concrete to isolated columns.

C/S Area (m2)	Aggregate Light	Dense	
< 0.03	.0	.0	(£/hr)
0.03 -0.10	7.1	.0	(£/hr)
0.10 -0.25	.0	.0	(£/hr)
> 0.25	.0	.0	(£/hr)

Please update the following values of material unit cost (£/m3).

Reinforced concrete to walls, columns, slabs,beams & staircases.

Light Aggregate	Dense Aggregate
70.1 (£/m3)	.0 (£/m3)

FORMWORK FOR STAIRCASES AND SLABS

Please update the values of material unit cost (muc) (£/m2) and labour unit cost (luc) (£/hr) for the following, if required.

Formwork for soffits of stairs not exceeding 200 mm thickness.

Height (m)	muc	luc
< 3.5	18.6	11.5

Formwork for soffits of landings not exceeding 200 mm thickness.

Height (m)	muc	luc
< 3.5	.0	.0

Formwork to soffits of slabs

Thickness (mm)	muc	luc
<200	.0	.0
200-300	18.6	11.3
300-400	.0	.0

Formwork to the vertical edges of slabs

Depth (mm)	muc (£/m)	luc
<250	.0	.0
250-500	.0	.0

REINFORCED CONCRETE FOR SLABS AND BEAMS

Please update the following values of unit labour cost (£/hr) of for reinforced concrete to slabs, isolated beams, deep beams and staircases.

SUSPENDED SLABS

Thickness (mm)	< 100	100 -150	150 -300	> 300
luc (£/hr)	.0	.0	7.1	.0

ISOLATED BEAMS

Cross Sectional Area (m2)	< 0.03	0.03 -0.10	0.10 -0.25	> 0.25
luc (£/hr)	.0	.0	.0	.0

DEEP BEAMS

Cross Sectional Area (m2)	0.03 -0.10	0.10 -0.25	> 0.25
luc (£/hr)	.0	.0	.0

STAIRCASES

luc (£/hr)	7.1

BLOCKWORK

Please update the following values of unit material cost (umc) (£/m2) and unit labour cost (ulc) (£/hr) for precast concrete blocks, (BS6073), compressive strength 7N/mm2, in gauged mortar.

	Walls and partitions					Skins of hollow blocks			
	Solid blocks		Hollow blocks			Solid blocks		Hollow blocks	
Thickness (mm)	umc (£/m2)	ulc (£/hr)	muc (£/m2)	luc (£/hr)	Thickness (mm)	umc (£/m2)	ulc (£/hr)	muc (£/m2)	luc (£/hr)
60	.0	.0			90	.0	.0		
75	.0	.0			115	.0	.0	.0	.0
100	.0	0	.0	.0	125	.0	.0	.0	.0
150	7.0	9.2	.0	.0	140	.0	.0	.0	.0
200	.0	.0	.0	.0	190	.0	.0	.0	.0
215			.0	.0					

Piers

	Solid blocks		Hollow blocks			
Thickness (mm)	umc	ulc	umc	ulc		
100	.0	.0	.0	.0		
150	.0	.0	.0	.0		
190	.0	.0	.0	.0		
200	.0	.0	.0	.0		
215	.0	.0	.0	.0	*muc	*luc
* Extra over unit prices for fair face and flush pointing					.0	.0

Please update the following values of unit material cost (umc) (£/m2) and unit labour cost (ulc) (£/hr) for Facing bricks, PC £160 per 1000 in gauged mortar (1:2:9)

	Thickness (mm) 102.5		215	
	umc	ulc	umc	ulc
- Walls	.0	.0	38.9	8.0
- Skins of hollow walls	.0	.0		

BLOCKWORK

Please update the following values of unit material cost (umc) (£/m2) and unit labour cost (ulc) (£/hr) for precast concrete blocks, (BS6073), compressive strength 7N/mm2, in gauged mortar.

	Walls and partitions					Skins of hollow blocks			
	Solid blocks		Hollow blocks			Solid blocks		Hollow blocks	
Thickness (mm)	umc (£/m2)	ulc (£/hr)	muc (£/m2)	luc (£/hr)	Thickness (mm)	umc (£/m2)	ulc (£/hr)	muc (£/m2)	luc (£/hr)
60	.0	.0			90	.0	.0		
75	.0	.0			115	.0	.0	.0	.0
100	7.0	9.2	.0	.0	125	.0	.0	.0	.0
150	.0	.0	.0	.0	140	.0	.0	.0	.0
200	.0	.0	.0	.0	190	.0	.0	.0	.0
215			.0	.0					

	Piers					
	Solid blocks		Hollow blocks			
Thickness (mm)	umc	ulc	umc	ulc		
100	.0	.0	.0	.0		
150	.0	.0	.0	.0		
190	.0	.0	.0	.0		
200	.0	.0	.0	.0		
215	.0	.0	.0	.0	*muc	*luc
* Extra over unit prices for fair face and flush pointing					.0	.0

Please input the quantıes and unit costs for Mastic Asphalt BS 988.
20 mm Two coat work rubbed with fine sand; to concrete base: flat coverings:

	Quantity	ulc (£/hr)	umc(£/m2)
over 300 mm wide	.0	.0	.0
not exceeding 150 mm wide	1200.0	7.0	7.0
150 - 300 mm wide	.0	.0	.0

APPENDIX 9 OUTPUT SCREENS DURING THE VALIDATION OF THE SIMULATION MODEL.

VALUES OF TIME AND COST FOR THE ACTIVITIES OF:-
- EXCAVATION
- EARTHWORK SUPPORT
- COMPACTION

ACTIVITY	COST(pound)	TIME (hr)	(day)	(week)
EXCAVATION FOR:				
Topsoil to be removed	216.0	10.3	1.3	.3
Topsoil to be preserved	108.0	5.1	.6	.1
Reduce level	428.4	20.4	2.5	.5
Basements	.0	.0	.0	.0
Trenches	283.5	13.5	1.7	.3
COMPACTION OF:				
The bottom of all excavations	.0	.0	.0	.0

VALUES FOR TIME AND COST FOR THE ACTIVITIES OF
REINFORCED CONCRETE FOUNDATIONS

ACTIVITY	COST(pound)	TIME (hr)	(day)	(week)
FORMWORK	1236.2	21.0	2.6	.5
REINFORCEMENT	7946.1	8.1	8.1	1.6
CONCRETE	27128.0	63.2	7.9	1.6
BLINDING LAYER	803.7	18.0	2.3	.4

VALUES OF TIME AND COST FOR THE ACTIVITIES OF:-
- FORMWORK FOR: COLUMNS AND WALLS, STAIRCASES, SLABS AND BEAMS.
- REINFORCEMENT FOR: COLUMNS AND WALLS, STAIRCASES, SLABS AND BEAMS.
- REINFORCED CONCRETE FOR: COLUMNS AND WALLS, STAIRCASES, SLABS AND BEAMS.

ACTIVITY	COST(pound)	TIME (hr)	(day)	(week)
FORMWORK FOR:-				
Columns and Walls	1099.8	13.8	1.8	.3
Slabs and Beams	29100.0	100.0	12.8	2.5
Staircases	263.0	2.3	1.7	.3
REINFORCEMENT FOR:-				
Columns and Walls	814.5	16.5	2.1	.4
Slabs and Beams	15264.0	96.8	11.9	2.4
Staircases	157.5	5.8	.6	.1
REINFORCED CONCRETE FOR:-				
Columns and Walls	1317.8	18.8	2.3	.5
Slabs and Beams	26002.8	27.0	3.5	.7
Staircases	380.2	.9	.8	.2

VALUES FOR TIME AND COST FOR THE ACTIVITIES OF INTERNAL BRICKWORK AND BLOCKWORK

	COST(pound)	TIME (hr)	(day)	(week)
BLOCKWORK	6031.2	42.0	5.3	1.0
BRICKWORK	20380.1	63.0	7.9	1.6

VALUES FOR TIME AND COST FOR THE ACTIVITIES OF INTERNAL BRICKWORK AND BLOCKWORK

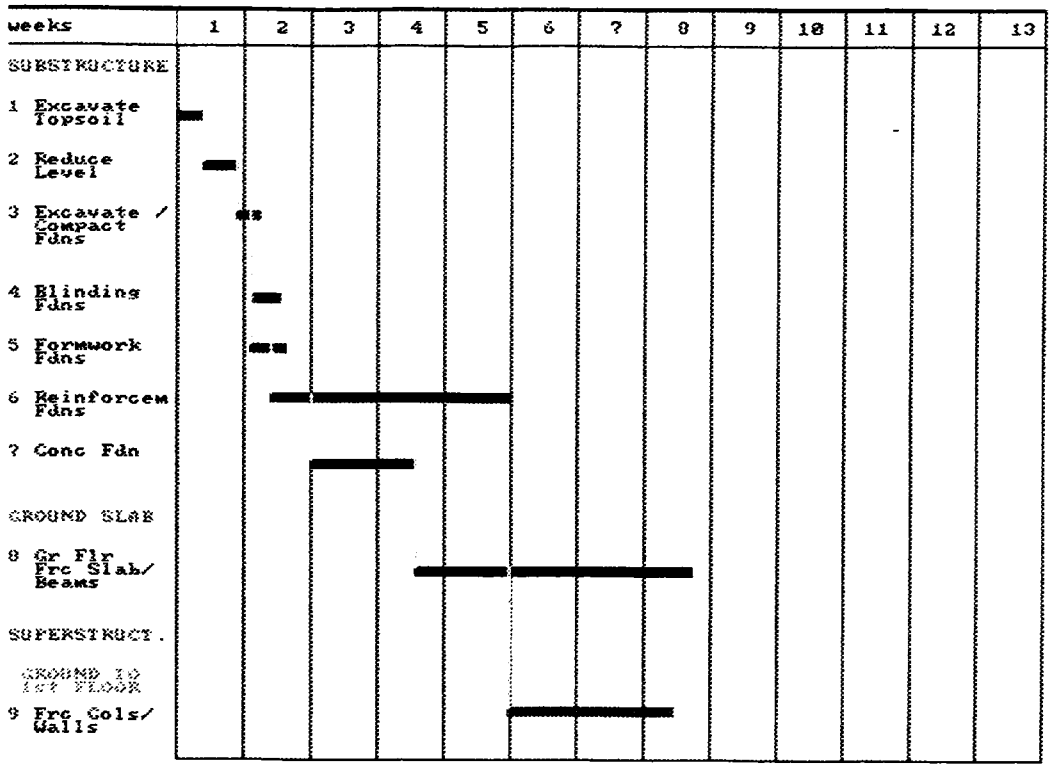
	COST(pound)	TIME (hr)	(day)	(week)
BLOCKWORK	2983.7	51.8	6.5	1.3
BRICKWORK	.0	.0	.0	.0

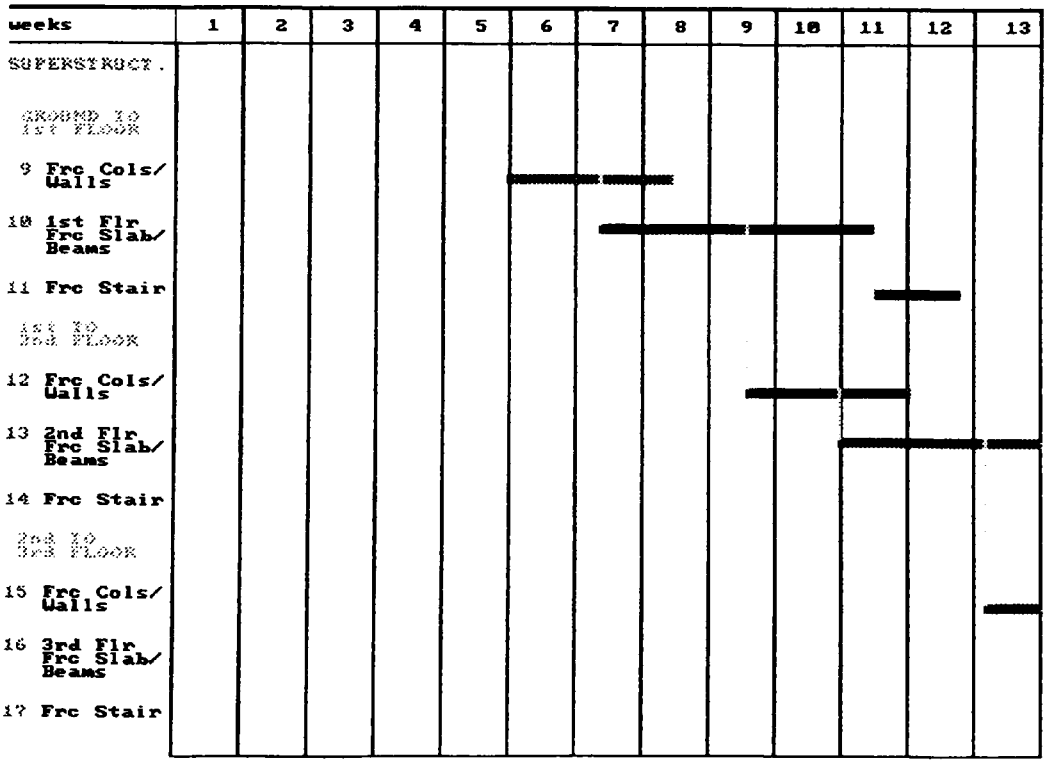
ROOF ASPHALT

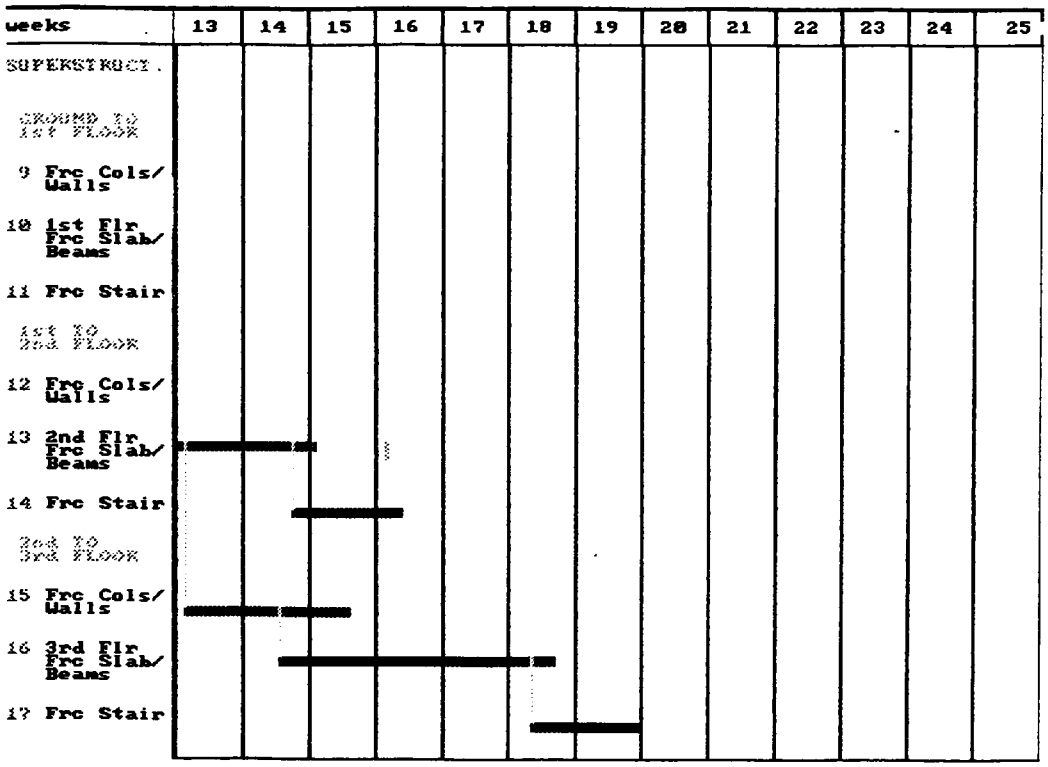
TIME AND COST VALUES OBTAINED ACCORDING TO NORMAL AND MAXIMUM GANG SIZES

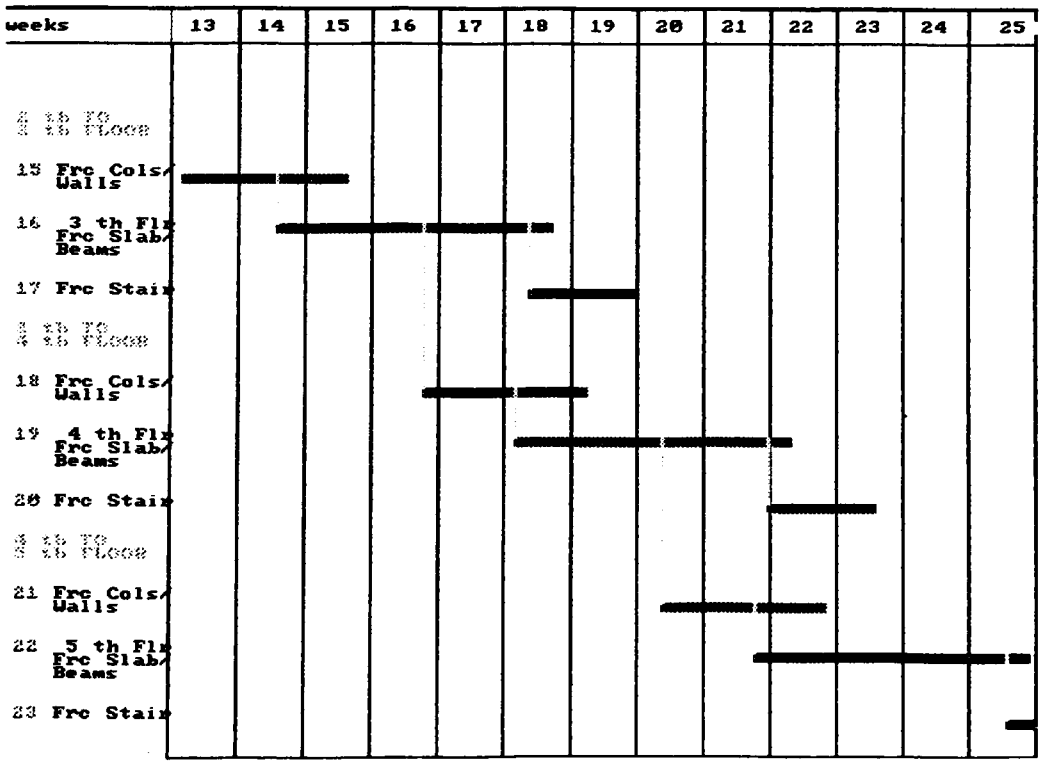
	Gang Size	Cost	Duration Hour	Day	Week
Normal	4	15120.0	240.0	30.0	6.0
Maximum	10	16387.6	114.4	14.3	2.8

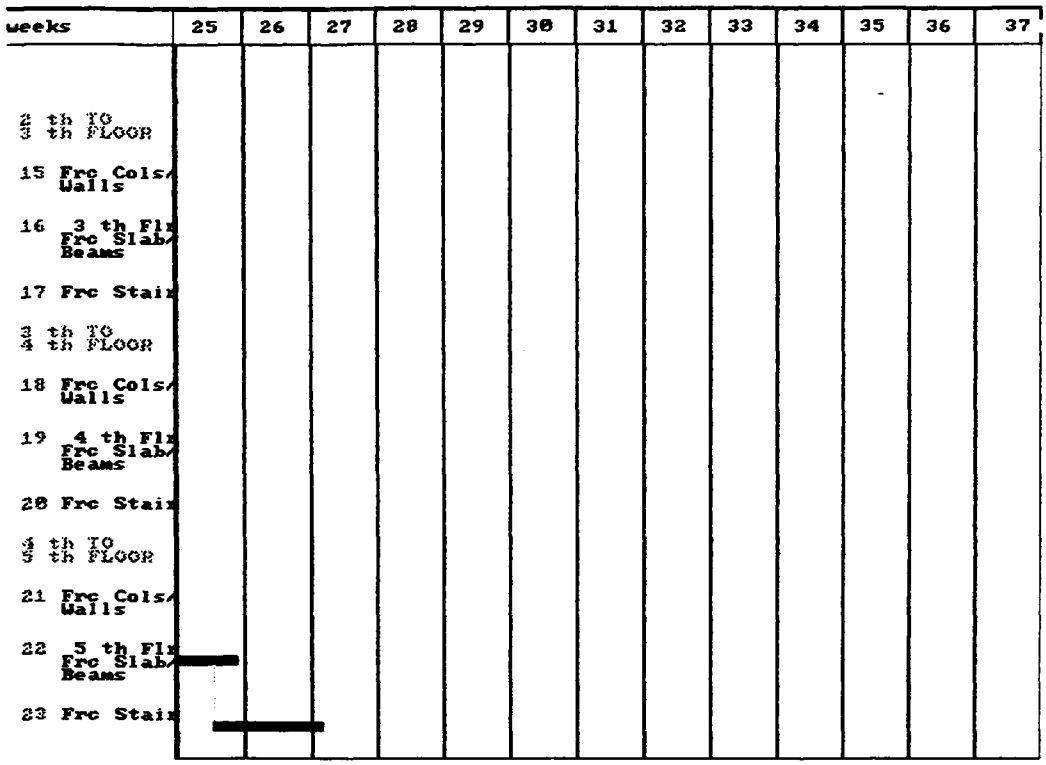
APPENDIX 10 THE BAR CHART DRAWN DURING THE VALIDATION OF THE SIMULATION MODEL.

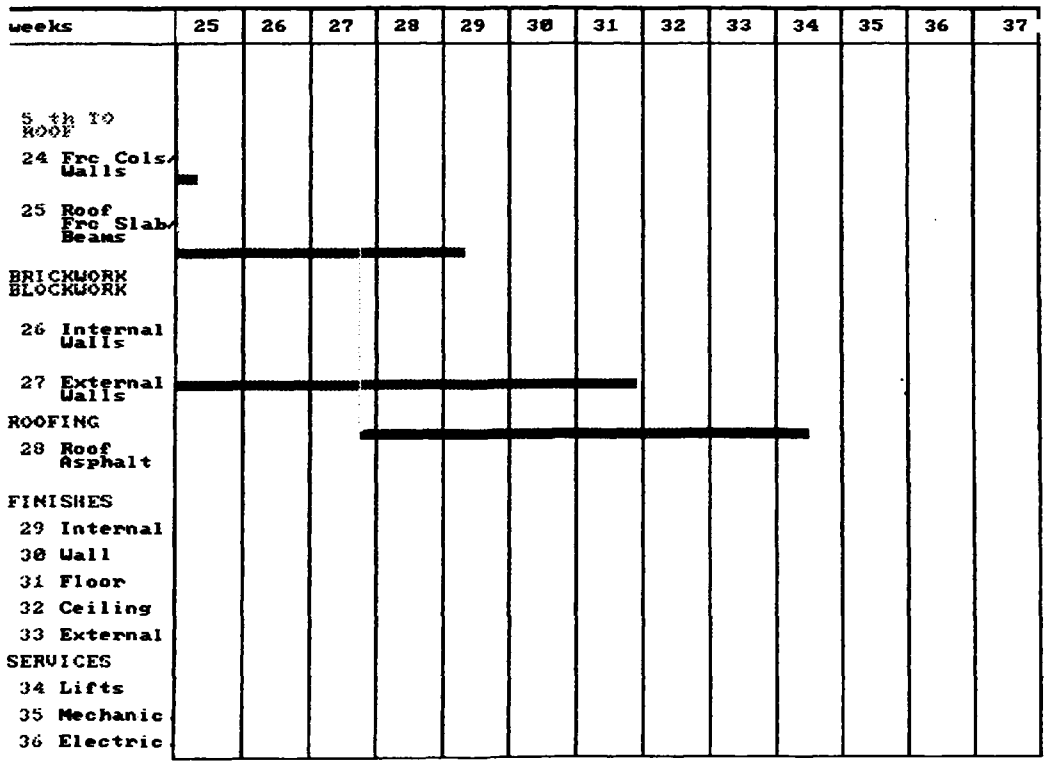

weeks
1
2
3
4
5
6
7
8
9
10
11
12
13
SUBSTRUCTURE
1 Excavate Topsoil
2 Reduce Level
3 Excavate / Compact Fdns
4 Blinding Fdns
5 Formwork Fdns
6 Reinforcem Fdns
7 Conc Fdn
GROUND SLAB
8 Gr Flr Frc Slab/ Beams
SUPERSTRUCT.
GROUND to 1ST FLOOR
9 Frc Cols/ Walls


weeks
1
2
3
4
5
6
7
8
9
10
11
12
13
SUPERSTRUCT.
GROUND to 1ST FLOOR
9 Frc Cols/ Walls
10 1st Flr Frc Slab/ Beams
11 Frc Stair
1st to 2nd FLOOR
12 Frc Cols/ Walls
13 2nd Flr Frc Slab/ Beams
14 Frc Stair
2nd to 3rd FLOOR
15 Frc Cols/ Walls
16 3rd Flr Frc Slab/ Beams
17 Frc Stair


weeks
13
14
15
16
17
18
19
20
21
22
23
24
25
SUPERSTRUCT.
GROUND TO 1st FLOOR
9 Frc Cols/ Walls
10 1st Flr Frc Slab/ Beams
11 Frc Stair
1st TO 2nd FLOOR
12 Frc Cols/ Walls
13 2nd Flr Frc Slab/ Beams
14 Frc Stair
2nd TO 3rd FLOOR
15 Frc Cols/ Walls
16 3rd Flr Frc Slab/ Beams
17 Frc Stair


weeks
13
14
15
16
17
18
19
20
21
22
23
24
25
4 th TO 5 th Floor
15 Frc Cols/ Walls
16 3 th Flr Frc Slab/ Beams
17 Frc Stair
4 th TO 5 th Floor
18 Frc Cols/ Walls
19 4 th Flr Frc Slab/ Beams
20 Frc Stair
4 th TO 5 th Floor
21 Frc Cols/ Walls
22 5 th Flr Frc Slab/ Beams
23 Frc Stair


weeks
25
26
27
28
29
30
31
32
33
34
35
36
37
2 th TO
3 th FLOOR
15 Frc Cols/
Walls
16 3 th Flr
Frc Slab/
Beams
17 Frc Stair
3 th TO
4 th FLOOR
18 Frc Cols/
Walls
19 4 th Flr
Frc Slab/
Beams
20 Frc Stair
4 th TO
5 th FLOOR
21 Frc Cols/
Walls
22 5 th Flr
Frc Slab/
Beams
23 Frc Stair


weeks
25
26
27
28
29
30
31
32
33
34
35
36
37
5 th TO
ROOF
24 Frc Cols/
Walls
25 Roof
Frc Slab/
Beams
BRICKWORK
BLOCKWORK
26 Internal
Walls
27 External
Walls
ROOFING
28 Roof
Asphalt
FINISHES
29 Internal
30 Wall
31 Floor
32 Ceiling
33 External
SERVICES
34 Lifts
35 Mechanic
36 Electric

APPENDIX 11 THE OUTPUT SCREENS FOR ACCELERATION OF THE PROJECT DURATION ACCELERATION BY EMPLOYING MORE LABOUR

	Normal DC	Dur	Amoount/Type of Plant Normal		Maximum		Values According to Max. Amount of Plant DC	Dur
Excavate topsoil to be removed/ preserved	324.0	1.9	1	1	2	6	385.7	1.0
Excavate to reduce levels	428.4	2.5	1	1	2	6	510.0	1.9
Excavate for basements	.0	.0	1	0	2	6	.0	.0
Excavate for trenches	283.5	1.7	1	1	2	6	337.0	.8
Compact the bottom of excavations	.0	.0	1	0	0	0	.0	.0

	Normal DC	Dur	Normal Gang Size	Maximum Gang Size	Values According to Maximum Gang Size DC	Dur
Formwork	1236.2	2.6	2	5	1332.3	1.2
Reinforc.	7946.1	8.1	3	4	8185.1	6.1
Concrete	27128.0	7.9	2	7	27802.1	3.2
Blinding	803.7	2.3	2	4	808.0	1.2

	Normal DC	Normal Dur	Normal Gang Size	Maximum Gang Size	Values According to Maximum Gang Size DC	Values According to Maximum Gang Size Dur
Formwork:-						
-Col/wall.	1099.8	1.7	4	8	1160.8	1.0
-Slab/beam.	29100.0	12.5	6	12	29766.1	6.9
-Staircase.	300.2	.5	2	4	312.3	.2
Reinfor.:-						
-Col/wall.	814.5	2.1	2	6	816.2	.8
-Slab/beam.	15264.0	12.0	6	9	15272.0	8.1
-Staircase.	157.5	.7	1	2	157.2	.4
Concrete:-						
-Col/wall.	1317.8	2.3	2	4	1329.6	1.3
-Slab/beam.	26002.8	3.4	4	8	26262.0	1.9
-Staircase.	267.7	.1	2	3	269.2	.1

	Normal Gang Size	Normal DC	Normal Dur	Maximum Gang Size	Values According to Max Gang Size DC	Values According to Max Gang Size Dur
Blockwork	8	6031.2	5.3	14	6321.1	3.2
Brickwork	8	20380.4	7.9	14	20622.0	4.8

	Normal Gang Size	Normal DC	Normal Dur	Maximum Gang Size	Values According to Max Gang Size DC	Values According to Max Gang Size Dur
Blockwork	3	2983.7	6.5	7	3255.3	3.1
Brickwork	0	0.0	0.0	0	0.0	0.0

ROOF ASPHALT

TIME AND COST VALUES OBTAINED ACCORDING TO NORMAL AND MAXIMUM GANG SIZES

	Gang Size	Cost	Duration Hour	Duration Day	Duration Week
Normal	4	15120.0	240.0	30.0	6.0
Maximum	10	16387.6	114.4	14.3	2.8

APPENDIX 12 OUTPUT FROM THE OPTIMISATION MODEL DURING THE VALIDATION

THE OPTIMISATION RESULTS FROM THE VALIDATION RUN

1st iteration:

optimal solution/objective function: 0.0

project duration is: 172 days

direct cost increase is: 0.0

indirect cost decrease is: 0.0

project cost increase is: 0.0

new direct project cost is: 676122.9

new indirect project cost is: 16900

new total project cost is: 693022.9

2nd iteration:

optimal solution/objective function: 1.7

project duration is: 171 days

direct cost increase is: 1.7

indirect cost decrease is: 98.256

project cost increase is: -96.556

new direct project cost is: 676124.6

new indirect project cost is: 16801.74

new total project cost is: 692926.3

activity to be accelerated : gr-1 floor / reinforcement for columns/walls

3rd iteration:

optimal solution/objective function: 3.4

project duration is: 170 days

direct cost increase is: 3.4

indirect cost decrease is: 196.512

project cost increase is: -193.112

new direct project cost is: 676126.3

new indirect project cost is: 16703.49

new total project cost is: 692829.8

activity to be accelerated : gr-1 / reinforcement for columns/walls

4th iteration:

optimal solution/objective function: 5.1

project duration is: 169 days

direct cost increase is: 5.1

indirect cost decrease is: 294.768

project cost increase is: -289.667

new direct project cost is: 676128.

new indirect project cost is: 16605.23

new total project cost is: 692733.2

activity to be accelerated : 1-2 / reinforcement for columns/walls

5th iteration:

optimal solution/objective function: 6.8

project duration is: 168 days

direct cost increase is: 6.8

indirect cost decrease is: 393.023

project cost increase is: -386.223

new direct project cost is: 676129.7

new indirect project cost is: 16506.98

new total project cost is: 692636.7

activity to be accelerated : 1-2 / reinforcement for columns/walls

6th iteration:

optimal solution/objective function: 8.5

project duration is: 167 days

direct cost increase is: 8.5

indirect cost decrease is: 491.278

project cost increase is: -482.779

new direct project cost is: 676131.4

new indirect project cost is: 16408.72

new total project cost is: 692540.1

activity to be accelerated : 2-3 / reinforcement for columns/walls

7th iteration:

optimal solution/objective function: 10.2

project duration is: 166 days

direct cost increase is: 10.2

indirect cost decrease is: 589.535

project cost increase is: -579.335

new direct project cost is: 676133.1

new indirect project cost is: 16310.47

new total project cost is: 692443.6

activity to be accelerated : 2-3 / reinforcement for columns/walls

8th iteration:

optimal solution/objective function: 11.9

project duration is: 165 days

direct cost increase is: 11.9

indirect cost decrease is: 687.791

project cost increase is: -675.891

new direct project cost is: 676134.8

new indirect project cost is: 16212.21

new total project cost is: 692347.

activity to be accelerated : 3-4 / reinforcement for columns/walls

9th iteration:

optimal solution/objective function: 13.6

project duration is: 164 days

direct cost increase is: 13.6

indirect cost decrease is: 786.047

project cost increase is: -772.447

new direct project cost is: 676136.5

new indirect project cost is: 16113.95

new total project cost is: 692250.5

activity to be accelerated : 3-4 / reinforcement for columns/walls

10th iteration:

optimal solution/objective function: 15.3

project duration is: 163 days

direct cost increase is: 15.3

indirect cost decrease is: 884.302

project cost increase is: -869.002

new direct project cost is: 676138.2

new indirect project cost is: 16015.7

new total project cost is: 692153.9

activity to be accelerated : 4-5 / reinforcement for columns/walls

11th iteration:

optimal solution/objective function: 17

project duration is: 162 days

direct cost increase is: 17

indirect cost decrease is: 982.558

project cost increase is: -965.558

new direct project cost is: 676139.9

new indirect project cost is: 15917.44

new total project cost is: 692057.3

activity to be accelerated : 4-5 / reinforcement for columns/walls

12th iteration:

optimal solution/objective function: 18.7

project duration is: 161 days

direct cost increase is: 18.7

indirect cost decrease is: 1080.814

project cost increase is: -1062.11

new direct project cost is: 676141.6

new indirect project cost is: 15819.19

new total project cost is: 691960.8

activity to be accelerated : 5-roof / reinforcement for columns/walls

13th iteration:

optimal solution/objective function: 20.4

project duration is: 160 days

direct cost increase is: 20.4

indirect cost decrease is: 1179.07

project cost increase is: -1158.67

new direct project cost is: 676143.3

new indirect project cost is: 15720.93

new total project cost is: 691864.2

activity to be accelerated : 5-roof / reinforcement for columns/walls

14th iteration:

optimal solution/objective function: 82.1

project duration is: 159 days

direct cost increase is: 82.1

indirect cost decrease is: 1277.326

project cost increase is: -1195.23

new direct project cost is: 676205.

new indirect project cost is: 15622.67

new total project cost is: 691827.7

activity to be accelerated : excavate topsoil

15th iteration:

optimal solution/objective function: 163.7

project duration is: 158 days

direct cost increase is: 163.7

indirect cost decrease is: 1375.581

project cost increase is: -1211.88

new direct project cost is: 676286.6

new indirect project cost is: 15524.42

new total project cost is: 691811.

activity to be accelerated : reduce levels

16th iteration:

optimal solution/objective function: 248.21

project duration is: 157 days

direct cost increase is: 248.21

indirect cost decrease is: 1473.837

project cost increase is: -1225.63

new direct project cost is: 676371.1

new indirect project cost is: 15426.16

new total project cost is: 691797.3

activity to be accelerated : roof asphalt

17th iteration:

optimal solution/objective function: 332.72

project duration is: 156 days

direct cost increase is: 332.72

indirect cost decrease is: 1572.093

project cost increase is: -1239.37

new direct project cost is: 676455.6

new indirect project cost is: 15327.91

new total project cost is: 691783.5

activity to be accelerated : roof asphalt

18th iteration:

optimal solution/objective function: 417.23

project duration is: 155 days

direct cost increase is: 417.23

indirect cost decrease is: 1670.349

project cost increase is: -1253.12

new direct project cost is: 676540.1

new indirect project cost is: 15229.65

new total project cost is: 691769.8

activity to be accelerated : roof asphalt

19th iteration:

optimal solution/objective function: 501.74

project duration is: 154 days

direct cost increase is: 501.74

indirect cost decrease is: 1768.605

project cost increase is: -1266.86

new direct project cost is: 676624.6

new indirect project cost is: 15131.4

new total project cost is: 691756

activity to be accelerated : roof asphalt

20th iteration:

optimal solution/objective function: 586.25

project duration is: 153 days

direct cost increase is: 586.25

indirect cost decrease is: 1866.86

project cost increase is: -1280.61

new direct project cost is: 676709.2

new indirect project cost is: 15033.14

new total project cost is: 691742.3

activity to be accelerated : roof asphalt

21st iteration:

optimal solution/objective function: 670.76

project duration is: 152 days

direct cost increase is: 670.76

indirect cost decrease is: 1965.116

project cost increase is: -1294.36

new direct project cost is: 676793.7

new indirect project cost is: 14934.88

new total project cost is: 691728.5

activity to be accelerated : roof asphalt

22nd iteration:

optimal solution/objective function: 755.27

project duration is: 151 days

direct cost increase is: 755.27

indirect cost decrease is: 2063.372

project cost increase is: -1308.1

new direct project cost is: 676878.2

new indirect project cost is: 14836.63

new total project cost is: 691714.8

activity to be accelerated : roof asphalt

23rd iteration:

optimal solution/objective function: 839.78

project duration is: 150 days

direct cost increase is: 839.78

indirect cost decrease is: 2161.628

project cost increase is: -1321.85

new direct project cost is: 676962.7

new indirect project cost is: 14738.37

new total project cost is: 691701.1

activity to be accelerated : roof asphalt

24th iteration:

optimal solution/objective function: 924.29

project duration is: 149 days

direct cost increase is: 924.29

indirect cost decrease is: 2259.884

project cost increase is: -1335.59

new direct project cost is: 677047.2

new indirect project cost is: 14640.12

new total project cost is: 691687.3

activity to be accelerated : roof asphalt

25th iteration:

optimal solution/objective function: 1008.8

project duration is: 148 days

direct cost increase is: 1008.8

indirect cost decrease is: 2358.14

project cost increase is: -1349.34

new direct project cost is: 677131.7

new indirect project cost is: 14541.86

new total project cost is: 691673.6

activity to be accelerated : roof asphalt

26th iteration:

optimal solution/objective function: 1093.31

project duration is: 147 days

direct cost increase is: 1093.31

indirect cost decrease is: 2456.395

project cost increase is: -1363.09

new direct project cost is: 677216.2

new indirect project cost is: 14443.6

new total project cost is: 691659.8

activity to be accelerated : roof asphalt

27th iteration:

optimal solution/objective function: 1177.82

project duration is: 146 days

direct cost increase is: 1177.82

indirect cost decrease is: 2554.651

project cost increase is: -1376.83

new direct project cost is: 677300.7

new indirect project cost is: 14345.35

new total project cost is: 691646.1

activity to be accelerated : roof asphalt

28th iteration:

optimal solution/objective function: 1262.33

project duration is: 145 days

direct cost increase is: 1262.33

indirect cost decrease is: 2652.907

project cost increase is: -1390.58

new direct project cost is: 677385.2

new indirect project cost is: 14247.09

new total project cost is: 691632.3

activity to be accelerated : roof asphalt

29th iteration:

optimal solution/objective function: 1346.84

project duration is: 144 days

direct cost increase is: 1346.84

indirect cost decrease is: 2751.163

project cost increase is: -1404.32

new direct project cost is: 677469.7

new indirect project cost is: 14148.84

new total project cost is: 691618.6

activity to be accelerated : roof asphalt

30th iteration:

optimal solution/objective function: 1431.35

project duration is: 143 days

direct cost increase is: 1431.35

indirect cost decrease is: 2849.419

project cost increase is: -1418.07

new direct project cost is: 677554.3

new indirect project cost is: 14050.58

new total project cost is: 691604.8

activity to be accelerated : roof asphalt

31st iteration:

optimal solution/objective function: 1560.95

project duration is: 142 days

direct cost increase is: 1560.95

indirect cost decrease is: 2947.674

project cost increase is: -1386.72

new direct project cost is: 677683.9

new indirect project cost is: 13952.33

new total project cost is: 691636.2

activity to be accelerated : concrete for roof slabs/beams

optimum duration is: 143 days

32nd iteration:

optimal solution/objective function: 1690.55

project duration is: 141 days

direct cost increase is: 1690.55

indirect cost decrease is: 3045.93

project cost increase is: -1355.38

new direct project cost is: 677813.5

new indirect project cost is: 13854.07

new total project cost is: 691667.5

activity to be accelerated : concrete for roof slabs/beams

33rd iteration:

optimal solution/objective function: 1825.35

project duration is: 140 days

direct cost increase is: 1825.35

indirect cost decrease is: 3144.19

project cost increase is: -1318.8

new direct project cost is: 677948.0

new indirect project cost is: 13755.8

new total project cost is: 691704.0

activity to be accelerated : formwork for foundations

34th iteration:

optimal solution/objective function: 1960.15

project duration is: 139 days

direct cost increase is: 1960.15

indirect cost decrease is: 3242.44

project cost increase is: -1282.3

new direct project cost is: 678083.0

new indirect project cost is: 13657.6

new total project cost is: 691741.0

activity to be accelerated : formwork for foundations

35th iteration:

optimal solution/objective function: 2094.95

project duration is: 138 days

direct cost increase is: 2094.95

indirect cost decrease is: 3340.7

project cost increase is: -1245.7

new direct project cost is: 678218.0

new indirect project cost is: 13559.3

new total project cost is: 691777.0

activity to be accelerated : formwork for foundations

36th iteration:

optimal solution/objective function: 2229.75

project duration is: 137 days

direct cost increase is: 2229.75

indirect cost decrease is: 3438.95

project cost increase is: -1209.2

new direct project cost is: 678353.0

new indirect project cost is: 13461.0

new total project cost is: 691814.0

activity to be accelerated : formwork for foundations

APPENDIX 13 CODING FOR THE TIME/COST MODEL

```
'MAIN MENU'

seek sta

use screen_start2

run file_startnext(act_start,filesn)

run start_escape()


'SITE SET UP'

seek site_set_up

run con_set2(sit)

run cost_set2(csit)

run con_seta2(sita)

run lag_mains2(lagmain)

run lag_mains2m(lagmainm)

run co_opta(sicoo)

run ntore5(new5)

run co_optfa(sopt)

if new5 is 'y' then
run sso(s86,s96,s137,s162,s187,s281,s368, s455, s464, s39, s40
s37,s42,s46,s47,s48,s49,s50,s51,s52,s54,s55,s56, s57, s58, s59,
s60,s62,s63,s64,s65,s66,s67,s68,s106,s107,s108,s109,s110, s111,
s112,s8,s20,s22,s24,s82,s133,s158,s183,s183, s461, s461, s158,
s461,lt_lr,lr_lb,lb_lt,lt_lc,s462,s44,s53,s61,s69,s40, s43, s9,
s10,s11,s12,s13,s14,s15,s16,s17,s18,s27,s28,s29,s30,s31,s32,
s33,s34,lt_lrm,lr_lbm,lb_ltm,lt_lcm)

if new5 is 'y' then
run ssol (sitotc,sdat,s896, s897, s897m, s138m, s163m, s188m,
s465m,s896m,s137m,s162m,s187m,s464m, sitotcm, sdatm, s8m, s20m,
s22m,s24m,s82m,s133m,s158m,s183m, s462m, s138, s163, s188,
s465,s137,s162,s187,s464,s8,s20,s22,s24, s82, s133, s158, s183,
s462)

use screen_4st

if st4 is '(1)' then
use screen4

if st4 is '(2)' then
use screen2

if st4 is '(3)' then
use screen1
```

```
if st4 is '(4)' then
use screen5

if st4 is '(5)' then
use screen_co_site

if st4 is '(7)' then
use screen_optbb

if st4 is '(8)' then
use screen_op

if st4 is '(9)' then
run st_return()

run s7(s8,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19)

run s7(s20,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s21)

run s7(s22,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s23)

run s7(s24,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s25)

run s7(s8m,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s19m)

run s7(s20m,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s21m)

run s7(s22m,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s23m)

run s7(s24m,s9,s10,s11,s12,s13,s14,s15,s16,s17,s18,s25m)

run s26(s27,s28,s29,s30,s31,s32,s33,s34,comp1,s36)

run s26(s27,s28,s29,s30,s31,s32,s33,s34,comp1m,s36m)

/*VOLUME OF EXCAVATED TOPSOIL TO BE REMOVED / PRESERVED

if s37>= 0 then
run volumes(s39,s37,s40)

if s41>=0 or s42>=0 or s40>=0 then
run s38(s41,s42,s43)

if s43>=0 then
s44=s40+s43

if s44 > 999999.9 then
s44=999999.9

if s44=0 then s8=0

/*VOLUME   OF    EXCAVATION TO REDUCE
LEVEL/BASEMENTS/TRENCHES

run vol(s46,s47,s48,s49,s50,s51,s52,s53)

run s45(s54,s55,s56,s57,s58,s59,s60,s61)
```

```
if s896 > 999999.9 then
s896=999999.9

if s897 > 999999.9 then
s897=999999.9

if s896m > 999999.9 then
s896m=999999.9

if s897m > 999999.9 then
s897m=999999.9

if s97 > 999999.9 then
s97=999999.9

if s97m > 999999.9 then
s97m=999999.9


/* REDUCE LEVEL/BASEMENTS/TRENCHES

run s99(s46,s47,s48,s49,s50,s51,s52,s54,s55,s56, s57, s58, s59,
s60,s62,s63,s64,s65,s66,s67,s68,s100,s101,s102,s103,s104,s105)

run s73(s70,s74)
/*REDUCE LEVEL

if s53>=0 and
s106>=0 and
s107>= 0 and
s108>=0 and
s109>=0 and
s110>=0 and
s111>=0 and
s112>=0 then
run s113(s106,s46,s114,s115);
run s113(s107,s47,s116,s117);
run s113(s108,s48,s118,s119);
run s113(s109,s49,s120,s121);
run s113(s110,s50,s122,s123);
run s113(s111,s51,s124,s125);
run s113(s112,s52,s126,s127);
s128=s115+s117+s119+s121+s123+s125+s127

if s128> 999999.9 then
s128=999999.9

if s114>=0
or s116>=0
or s118>=0
or s120>=0
or s122>=0
or s124>=0
or s126>=0
then run s78(s20,s1290,s1300);
run s78(s20m,s1290m,s1300m)

if s128>=0
```

```
then run s131(s46,s47,s48,s49,s50,s51,s52,s114,s116,s118,
S120,s122,s124,s126,s115,s117,s119,s121,s123,s125,s127,s132)
if s128=0 then s132=0
if s128>=0 then
run s81(s53,s1290,s101,s74,s132,s1300,s133,s21,snor2,s134,
s135,s136, s137);
s138=s136/8;
s139=s138/5;
s81 (s53, s1290m, s101, s74, s132, s1300m, s133m, s21m, snor2,
s134m,s135m,s136m,s137m);
s138m=s136m/8


/* BASEMENTS

if s61>=0 then
run s113(s106,s54,s140,s141);
run s113(s107,s55,s142,s143) ;
run s113(s108,s56,s144,s145)  ;
run s113(s109,s57,s146,s147);
run s113(s110,s58,s148,s149) ;
run s113(s111,s59,s150,s151);
run s113(s112,s60,s152,s153) ;
s154=s141+s143+s145+s147+s149+s151+s153

if s154 >999999.9 then
s154 = 999999.9
if s140>=0
or s142>=0
or s144>=0
or s146>=0
or s148>=0
or s150>=0
or s152>=0
then run s78(s22,s155,s156);
run s78(s22m,s155m,s156m)

if s154>=0 then
run s131(s54,s55,s56,s57,s58,s59,s60,s140,s142,s144,s146, s148,
s150,s152,s141,s143,s145,s147,s149,s151,s153,s157)

if s154=0 then s157=0

if s154 >=0 then
run s81(s61,s155,s103,s74,s157,s156,s158,s23,snor3, s159, s160,
s161,s162);
s163=s161/8;
s164=s163/5;
run s81 (s61, s155m, s103, s74, s157, s156m, s158m, s23m,snor3,
s159m,s160m,s161m,s162m);
s163m=s161m/8

/* TRENCHES

if s69>=0 then
run s113(s106,s52,s165,s166);
run s113(s107,s53,s167,s168) ;
run s113(s108,s54,s169,s170) ;
```

```
run s113(s109,s55,s171,s172);
run s113(s110,s56,s173,s174) ;
run s113(s111,s57,s175,s176) ;
run s113(s112,s58,s177,s178) ;
s179=s166+s168+s170+s172+s174+s176+s178

if s179 > 999999.9 then
s179=999999.9

if s165>=0
or s167>=0
or s169>=0
or s171>=0
or s173>=0
or s175>=0
or s177>=0
then run s78(s24,s180,s181);
run s78(s24m,s180m,s181m)

if s179>=0 then
run s131(s62,s63,s64,s65, s66, s67, s68, s165, s167, s169,s171,
s173, s175,s177,s166,s168,s170,s172,s174,s176,s178,s182)

if s179>=0 or s183>=0 then
run s81(s69,s180,s105,s74,s182,s181,s183,s25,snor4, s184, s185,
s186, s187);
s188=s186/8;
s189=s188/5;
run s81(s69,s180m,s105,s74,s182,s181m,s183m,s25m, snor4, s184m,
s185m,s186m,s187m);
s188m=s186m/8

/* COMPACT BOTTOMS OF EXCAVATIONS

run s458(comp1,s459)

run s460(s461,s459,s462,s36,snor5,s463,s464)

run s458(comp1m,s459m)

run s460(s461,s459m,s462m,s36,snor5,s463m,s464m)

if s463>= 0 then
s465=s463/8;
s466=s465/5;
s465m=s463m/8;
lb_lt=0;
lt_lc=0

if lt_lr >=0 or lr_lb>=0 or lb_lt>=0 or lt_lc>=0 then
run stt(s897,s138,s163,s188,s465,lt_lr,lr_lb,lb_lt,lt_lc, sdat,
shrt,swet,Send);

if lt_lrm >=0 or lr_lbm>=0 or lb_ltm>=0 or lt_lcm>=0 then
run stt (s897m, s138m, s163m, s188m, s465m,lt_lrm,lr_lbm,
lb_ltm,lt_lcm,sdatm,shrtm,swetm,Sendm)
```

```
if st4 is '(6)' then
use screen_finall
run co_sit(s86,s96,s137,s162,s187,s464,sitotc)

run co_sit(s86m,s96m,s137m,s162m,s187m,s464m,sitotcm)

run con_set(s39,s40,s37,s42,s46,s47,s48,s49,s50, s51, s52, s54,
s55,s56,s57, s58, s59, s60,s62, s63, s64, s65, s66, s67,
s68,s106, s107, s108,s109, s110, s111, s141, s112, s8, s20,
s22, s24, s82, s133,s158,s183,s766,s70,sit)

run con_seta (s461,lt_lr,lr_lb, lb_lt, lt_lc, s462, s44, s53,
s61,s69,s40,s43,comp1,sita)

run cost_set(s9,s10,s11,s12,s13,s14,s15,s16,s17, s18, s27, s28,
s29,s30,s31,s32,s33,s34,csit)
run co_opt(sitotc,sdat,sitotcm, sdatm, s897m, s138m, s163m,
s188m,s465m,s896m,s137m,s162m,s187m,s464m,s896,s897, s8m, s20m,
s22m,s24m,s82m,s133m,s158m,s183m,s462m,s138, s163, s188, s465,
s137,s162,s187,s464,s8,s20,s22,s24,s82,s133,s158,s183,s462,
comp1,comp1m,sicoo)
run co_optf (sdat,sdatm,Send,Sendm,snor1, snor2, snor3, snor4,
snor5, sopt)

run finsset (s88,s98,s139,s164,s189,s466,lt_lr,lr_lb,lb_lt,
lt_lc,gra1)

if st4 is not '(9)' then
run set_return()

run dummy2(new_project)

run nstore5(new_project,new5)

run exc_escape()
```

CONCRETE FOUNDATIONS'

```
seek foundation_concrete

run lag_mains2(lagmain)

run lag_mains2m(lagmainm)

run con_fou2(ffo)

run cost_form2(cform)

run cost_fou2(cffo)

run co_opt2 (coofo)

run ntore4(new4)

run co_optf2(fopt)
```

```
if new4 is 'y' or new4 is 'n' or lag_sifo>=0 or f1_f2>=0 or
f2_f3>=0 or lag_sifom>=0 or f1_f2m>=0 or f2_f3m>=0 then

lt_lr=0;
lr_lb=0;
lb_lt=0;
lt_lc=0;
lt_lrm=0;
lr_lbm=0;
lb_ltm=0;
lt_lcm=0;
f129 =0;
f130= 0 ;
f131 =0  ;
f132= 0 ;
f133 =0 ;
f134= 0 ;
f135 =0 ;
f136= 0 ;
f137 =0 ;
f208= 0 ;
f209 =0 ;
f210= 0 ;
f211 =0 ;
f212= 0 ;
f213 =0 ;
f214= 0 ;
f215= 0 ;
ac=0

if new4 is 'y' then
run cono_fou(f6,f7,f8,f9a, f18, f19, f20, f21, f29, f30, f31,
f32,f40,f49,f50,f51,f52,f61,f62,f63,f64,f72,f73,f74,f75, f83,
f84,f85,f86,f94,f95,f96,f97,f105,f114,f115, f116, f117, f118,
f119,f120,f121,f122,f123,f129,f130,f131,f132,f133,f134, f135,
f136,f141,f142,f143,f144,f145,f146,f160,f161,f162,f163, f166,
f167,f168,f169,f172,f175,f176,f177,f178,f14, f14, f125, f180,
f180,f44,f109,f155,f181,f449,f45,f110,f156, f182, f182, f461,
f157,f183,f183,f1_f2,f2_f3,f2_f3);


run cono_pro(f125m,f14m,f241,f238, f2394, f451, f241m, f238m,
f2394m,f182m,f156m,f451m,cofototm, fodatm, f180m, f14b,f14bm,
f15,f15m,f451b,f451bm,f461b,f16,f16m,f1_f2m,f2_f3m)

use screen_2st

if st2 is '(1)' then
use screen21

if st2 is '(2)' then
use screen21c

if st2 is '(3)' then
use screen_gangsize2
```

```
if st2 is '(4)' then
use screen_ac

if st2 is '(6)' then
use screen_optfo

if st2 is '(7)' then
use screen_out

if st2 is '(8)' then
run st_return

if ac = 1 then
use screen_ov;

if ac=2 then
use screen_gangsize2;
use screen_lab


run f5(f6,f7,f8,f9a,f10,f11,f12,f13,fnor1,f14b, f194, ac,whb,
whlb,f14bm,glb,glb1,ffm,f15,f16,f451bm,f16m)


run f17(f18,f19,f20,f21,f22,f23,f24,f25,fnor4,f26,f27)

run f28(f29,f30,f31,f32,f33,f34,f35,f36,fnor41,f37, f38)

run f39(f40,f41,fnor2,f42,f43)

/* TOTAL TIME FOR PLAIN CONCRETE (gangsize is taken into
account here)

if f14=0 then
f44=0;
f45=0;
f46=0

if f14m=0 then
f45m=0

if f14>0 then
f47=f26+f37+f42;
f44=f47/f14;
f45= f44/8;
f46=f45/5

if ac=1 and f14>0 then
f47=f15+f26+f37+f42;
wh3=whlc*whc/100-0.40*whlc;
f45m=f47/f14/8*40/whc+wh3

if ac=2 then
f47=f15+f26+f37+f42;
ffm3=f47/f14m/8;
run Aa(f14m,f14,ffm3,glc,glc1,f45m,Ra3)
if f47> 999999.9 then
```

```
f47=999999.9
if f44> 999999.9 then
f44=999999.9

if f45> 999999.9 then
f45=999999.9

if f46> 999999.9 then
f46=999999.9

if f1_f2 >=0 or f2_f3>=0 or f1_f2m>=0 or f2_f3m>=0 then
run f48(f49,f50,f51,f52,f53,f54,f55,f56,fnor3,f58,f59)

run f60(f61,f62,f63,f64,f65,f66,f67,f68,fnor4,f69,f70)

run f71(f72,f73,f74,f75,f76,f77,f78,f79,fnor42,f80, f81)

run f82(f83,f84,f85,f86,f87,f88,f89,f90,fnor41,f91, f92)

run f93(f94,f95,f96,f97,f98,f99,f100,f101,fnor43, f102, f103)

run f104(f105,f106,fnor2,f107,f108)

/*TOTAL TIME FOR REINFORCED CONCRETE

if f14=0 then
f109=0;
f110=0;
f111=0

if f14m=0 then
f110m=0

if f14>0 then
f112=f58+f69+f80+f91+f102+f107;
f109=f112/f14;
f110=f109/8;
f111=f110/5

if ac=1 and f14>0 then
f112=f58+f69+f80+f91+f102+f107;
wh3=whlc*whc/100-0.40*whlc;
f110m=f112/f14/8*40/whc+wh3

if f110m <0 then
f110m=0



if ac=2 then
f112=f58+f69+f80+f91+f102+f107;
ffm4=f112/f14m/8;
run Aa(f14m,f14,ffm4,glc,glc1,f110m,Ra4)


if f15>=0 then
f451b=f15/8;
```

```
f46lb=f15/5/8
if f109 > 999999.9 then
f109=999999.9

if f110 > 999999.9 then
f110=999999.9

if f111 > 999999.9 then
f111=999999.9

if f112 > 999999.9 then
f112=999999.9

if f15 >999999.9 then
f15=999999.9

if f4499 >999999.9 then
f4499=999999.9

run f113(f114,f115,f116,f117,f118,f119,f120,f121,f122,f123,
f124,fnor5,f126,f127)


run f140(f141,f142,f143,f144,f145,f146,f147,f148,f149,
f150,f151,f152,fnor6,f153,f154)


/*TOTAL TIME FOR REINFORCEMENT

if f125=0 then
f155=0;
f156=0;
f157=0

if f125m=0 then
f156m=0

if f125>0 then
f1588=f126+f153;
f155=f1588/f125;
f156=f155/8;
f157=f156/5

if ac=1 and f125>0 then
f1588=f126+f153;
whl=whlr*whr/100-0.40*whlr;
f156m=f1588/f125/8*40/whc+whl

if f156m<0 then
f156m=0



if ac=2 then
f1588=f126+f153;
ffm1=f1588/f125m/8;
run Aa(f125m,f125,ffm1,glr,glr1,f156m,Ra1)
```

```
run f159(f160,f161,f162,f163,lfm1,lfm2,lfm3,lfm4,fnor7,
f164,cf164)

run f165(f166,f167,f168,f169,lfm5,lfm6,lfm7,lfm8,fnor7,
f170,cf170)

run f171(f172,lfm9,fnor7,f173,cf173)

run f174(f175,f176,f177,f178,lfm10,lfm11,lfm12,lfm13,fnor7,
f179,cf179)
 :

/*TOTAL TIME FOR FORMWORK

if f180=0 then
f181=0

if f180>0 then
f1844=f164+f170+f173+f179;
f181=f1844/f180;
f182=f181/8;
f183=f182/5

if ac=1 and f180>0 then
f1844=f164+f170+f173+f179;
wh=whlf*whf/100-0.40*whlf;
f182m=f1844/f180/8*40/whf+wh

if f182m<0 then
f182m=0

if ac=2 then
f1844=f164+f170+f173+f179;
ffm=f1844/f180m/8;
run Aa(f180m,f180,ffm,glf,glf1,f182m,Ra)

if f1844>999999.9 then
f1844=999999.9

if f184 > 999999.9 then
f184=999999.9

if f1844m>999999.9 then
f1844m=999999.9

if f184m> 999999.9 then
f184m=999999.9

if f1588>999999.9 then
f1588=999999.9

if f158 > 999999.9 then
f158=999999.9

if f1588m>999999.9 then
f1588m=999999.9
```

```
if f158m> 999999.9 then
f158m=999999.9

if f181 > 999999.9 then
f181=999999.9

if f182 > 999999.9 then
f182=999999.9

if f183 > 999999.9 then
f183=999999.9

if f182m > 999999.9 then
f182m=999999.9

if f155 > 999999.9 then
f155=999999.9

if f156 > 999999.9 then
f156=999999.9

if f157 > 999999.9 then
f157=999999.9

if f156m > 999999.9 then
f156m=999999.9

if f44>=0 then
f449=f109+f44;
f451=f449/8;
f461=f451/5

if f449 > 999999.9 then
f449=999999.9

if f451 > 999999.9 then
f451=999999.9

if f461 > 999999.9 then
f461=999999.9

if f15 > 999999.9 then
f15=999999.9

if f451b > 999999.9 then
f451b=999999.9

if f461b > 999999.9 then
f461b=999999.9

if f451bm > 999999.9 then
f451bm=999999.9

if f110m >= 0 then
f451m=f110m+f45m

if f451m > 999999.9 then
f451m=999999.9
```

```
run f189(f45,f110,f156,f182,f191)

run f189(f46,f111,f157,f183,f192)

run f193(f6,f7,f8,f9a,f18,f19,f20,f21,f29,f30, f31, f32, f40,
f49,f50,f51,f52,f61,f62,f63,f64,f72,f73,f74,f75,f83, f84,f85,
f86,f94,f95,f96,f97,f105,f194,f195,f196)


run f197(f114,f115, f116, f117, f118, f119, f120, f121, f122,
f123,f141,f142,f143,f144,f145,f146,f160,f161,f162,f163,f166,
f167,f168,f169,f172,f175,f176,f177, f178,f198,f199,f200,
f201,f202,f203,f204,f205,f206,f207,f216,f217,f218,f219,f220,
f221,f222,f223,f224, f225, f226, f227, f228, f229, f230,
f231, f232,f233,f234,f235,f236)

run f237(f235, f127, f154, f195, f27, f38,f43, f196, f59,f70,
f92,f81,f103,f108,ac,f14m,f14,wh1,wh3, f125m,f125, glr, glr1,
glc,glc1,f238,f2394,f2394m,f238m)


run fcoform(cf164,cf170,cf173, cf179, f236, whf, ac, wh,f180,
f180m, glf,glf1,f241,f241m)


if lag_sifo>=0 then
run fott(lag_sifo,f182,f156, f451, f1_f2, f2_f3, fodat,fohrt,
fowet,Fend);

if lag_sifom>=0 then
run fott(lag_sifom,f182m,f156m,f451m, f1_f2m, f2_f3m, fodatm,
fohrtm,fowetm,Fendm)


if st2 is '(1)' or
st2 is '(2)' or
st2 is '(3)' or
st2 is '(6)' then
run found_return()
:

if st2 is '(4)' then
use screen_optfo;
run found_return()

if st2 is '(5)' then

use screen_final2;
run found_return()

run cototfou(f241,f238,f2394,f16,cofotot)

run cototfou(f241m,f238m,f2394m,f16m,cofototm)

run con_fou(f6,f7, f8, f9a, f18, f19, f20, f21, f29, f30,f31,
f32,f40,f49,f50,f51,f52,f61,f62,f63,f64,f72,f73,f74, f75,f83,
f84,f85, f86, f94, f95,f96,f97, f105, f114, f115, f116, f117,
f118, f119, f120,f121,f122,f123,f129, f130, f131, f132, f133,
```

```
f134, f135,f136,f141,f142,f143,f144,f145,f146,f160,f161,f162,
f163,f166,f167,f168,f169,f172,f175,f176,f177, f178, f14, f14,
f125,f180,f180,f44,f109,f155,f181,f181,f45,f110,f156,f182,
f182,f461,f157, f183,f183,f449,ffo)

run cost_fou(f10,f11,f12,f13,f22,f23,f24, f25, f33, f34, f35,
f36,f41,f53,f54,f55,f56,f65,f66,f67,f68,f76,f77,f78,f79,f87,
f88,f89,f90,f98,f99,f100,f101,f106,f124,f137,f147,f148,f149,
f150,f151,f152,f194,f198,f199,f200,f201,f202,f203,f204,f205,
f206,f207,f208,f209,f210,f211,f212,f213,f214,f215,f216,f217,
f218,f219,f220,f221,f222,f223,f224,f225,f226,f227,f228,f229,
f230,f231,f232,f233,f234,cofotot,cffo)

run cost_form(lfm1,lfm2,lfm3,lfm4,lfm5,lfm6, lfm7, lfm8,lfm9,
lfm10, lfm11,lfm12,lfm13,whf,whr,whc,whb,whlf,whlr,whlc,whlb,
glf,glr,glc,glb,glf1,glr1,glc1,glb1,cform)

run co_opt( cofotot, fodat, cofototm, fodatm, f2394m, f182m,
f156m, f451m, f241, f238, f2394,f451,f241m,f238m,f180m,f125m,
f14m, f14b, f14bm, f16,f16m,f15,f451b,f451bm,f461b,f182,f156,
coofo)

run co_optf(fodat,fodatm,Fend,Fendm,fnor1,fnor2,fnor3, fnor4,
fnor41, fnor42,fnor43,fnor5,fnor6,fnor7,fopt)


run fin_found(f183,f157,f461,f1_f2,f2_f3,f461b,foungr)

run dummy2(new_project)

run nstore4(new_project,new4)

run foun_escape()
```

```
'TYPICAL FLOOR'

seek typical_floor

run lag_mains2(lagmain)

run lag_mains2m(lagmainm)

run con_tyf2(typ)

run con_tyfa2(typa)

run num3a(noo3)

run ora2(unor)

run cost_typ2(ctyp)

run cost_typa2(ctypa)

run co_opt2(fltcoos)

run ntorel(new1)

if new1 is 'y' or new1 is 'n' then
tf166 =0;
tf167 =0;
tf168 =0;
tf169 =0;
tf170 =0;
tf171 =0;
tf172 =0;
tf173 =0;
tf266 =0;
tf267 =0;
tf268 =0;
tf269 =0;
tf270 =0;
tf271 =0;
tf272 =0;
tf273 =0;
tf274 =0;
tf275 =0;
tf276 =0;
tf277 =0;
tf278 =0;
tf279 =0;
tf280 =0;
tf281 =0;
tf52 =0;
tf53 =0;
tf54 =0;
tf55 =0;
tf56 =0;
tf57 =0;
tf58 =0;
tf59 =0;
tf60 =0;
```

```
tf61 =0;
tf62 =0;
tf63 =0;
tf64 =0;
tf65 =0;
tf66 =0;
tf69 =0;
tf331 =0;
tf332 =0;
tf333 =0;
tf334 =0;
tf335 =0;
tf336 =0;
tf337 =0;
tf338 =0;
tf311 =0;
tf137 =0;
tf139 =0;
tf343 =0;
tf143 =0;
tf345 =0;
tf145 =0;

if new1 is 'y' or new1 is 'n' then
lt_lr=0;
lr_lb=0;
lb_lt=0;
lt_lc=0;
f1_f2=0;
f2_f3=0;
lt_lrm=0;
lr_lbm=0;
lb_ltm=0;
lt_lcm=0;
f1_f2m=0;
f2_f3m=0;
lag_sifo=0;
lag_fogr=0;
lag_grfir=0;
lag_sifom=0;
lag_fogrm=0;
lag_grfirm=0

if new1 is 'y' then
run con_tyfk(tf8,tf9,tf10,tf11,tf12,tf13,tf14,tf15,tf16,tf32,
tf33,tf34,tf35,tf36,tf37,tf38,tf39,tf40,tf41,tf42,tf43, tf44,
tf45,tf46,tf47,tf48,tf49,tf50,tf51,tf52,tf53,tf54,tf55,tf56,
tf57,tf58,tf59,tf60,tf61,tf62,tf63,tf64,tf65,tf66,tf75,tf76,
tf77,tf78,tf79,tf80,tf81,tf82,tf83,tf84,tf85,tf86,tf87,tf88,
tf89,tf90,tf136,tf137,tf138,tf139,tf118, tf119, tf156, tf157,
tf158,tf159,tf160,tf161,tf162,tf163,tf164,tf165,tf166,tf167,
tf168,tf169,tf170,tf171,tf172,tf173,tf180,tf1810,tf182,tf183,
tf184,tf185,tf186,tf187,tf311,no3)


if new1 is 'y'  then
run con_tyfak(tf17,tf67,tf91,tf140,tf174,tf242, tf282, tf290,
tf141,tf196,tf197,tf198,tf199,tf200,tf201,tf202,tf203,tf204,
```

```
tf205,tf206, tf207, tf219, tf220, tf221, tf222, tf223,tf224,
tf225,tf226,tf227,tf228,tf229,tf246,tf247,tf248,tf249,tf250,
tf251,tf252,tf253,tf254,tf255,tf256,tf257,tf258,tf259,tf260,
tf261,tf262,tf263,tf264,tf265,tf266,tf267,tf268,tf269,tf270,
tf271,tf272,tf273,tf274,tf275,tf276,tf277,tf278,tf279,tf280,
tf281,tf122,tf123,tf124,tf125,tf126,tf127,tf128,tf129,tf130,
tf131,tf132,lag_cw1,lag_cw2,lag_sb1,lag_sb2,lag_sc1,lag_sc2,
lag_cwsb1, lag_sbs1, lag_cw1m, lag_cw2m, lag_sb1m, lag_sb2m,
lag_sc1m,lag_sc2m, lag_cwsb1m,lag_sbs1m)


if new1 is 'y' then
run co_fld(fltcos,fdat,tf373,tf374,tf375,tf376, tf377, tf378,
tf379,tf380,tf381,tf28,tf149,tf243, tf71, tf176, tf286,tf111,
tf150,tf303,tf373m,tf374m,tf375m,tf376m,tf377m,tf378m,tf379m,
tf380m,tf381m,tf28m,tf149m,tf243m,tf71m,tf176m,tf286m,tf111m,
tf150m,tf303m,fltcosm,fdatm,tf17m,tf140m,tf242m,tf67m,tf174m,
tf282m,tf91m,tf141m,tf290m)

use screen_3st

if st3 is '(0)' then
use screen_no3;
ac=3

if st3 is '(1)' then
use screen31;
ac=3

if st3 is '(2)' then
use screen31c;

ac=3

if st3 is '(3)' then
use screen_gangsize;
ac=3

if st3 is '(4)' then
use screen_ac

if st3 is '(6)' then
use screen_ac;
acc=1

if st3 is '(7)' then
use screen_or;
ac=3

if st3 is '(8)' then
ac=3;
run st_return

if ac =1 and
acc = 1 then
use screen_ov;
```

```
if ac =2 and
acc=1 then
use screen_gangsize;
use screen_lab

if st3 is '(5)' then
ac=3
run tf7(tf8,tf9,tf10,tf11,tf12,tf13,tf14,tf15,tf16,tf17,tf18,
tf19,tf20,tf21,tf22,tf23,tf24,tf25,tf26, no3, unor3, ac, whf,
whlf,tf17m,glf,glf1,tf27,tf28,tf29,wh1,tf28m)

if tf28 >= 0 then
tf30=tf28/5

if tf30 > 999999.9 then
tf30=999999.9
run tf31(tf32,tf33,tf34,tf35,tf36,tf37,tf38,tf39, tf40, tf41,
tf42,tf43,tf44,tf45,tf46,tf47,tf48,tf49,tf50,tf51,tf67, tf68,
no3,unor2,ac,whr,whlr,tf67m,glr,glr1,tf70, tf71, tf72, tf71m,
wh2)

if tf71 >=0 then
tf73=tf71/5

if tf73 > 999999.9 then
tf73=999999.9
run
tf74(tf75,tf76,tf77,tf78,tf79,tf80,tf81,tf82,tf83,tf84,tf85,
tf86,tf87,tf88,tf89,tf90,tf91,tf92,tf93,tf94,tf95,tf96, tf97,
tf98,tf99,tf100,tf101,tf102,tf103,tf104,tf105,tf106,tf107,no3
,unor1,ac,whc,whlc,tf91m,glc,glc1,tf108,tf109,tf110,tf111,
tf112,tf111m,wh3)

run
tf113(tf75,tf76,tf77,tf78,tf79,tf80,tf81,tf82,tf83,tf84,tf85,
tf86, tf87,tf88,tf89,tf90,tf114,tf115,no3,tf116)


run tf117(tf118,tf119,tf114,tf115,no3,tf120)

210:run
tf121(tf122,tf123,tf124,tf125,tf126,tf127,tf128,tf129,tf130,
tf131, tf132,tf114,tf115,no3,tf133)


if tf111 >=0 then
tf134=tf111/5

if tf134 > 999999.9 then
tf134=999999.9
run
tf135(tf136,tf137,tf138,tf139,tf118,tf119,tf140,tf141,tf142,
tf143, tf144,tf145,tf146,no3,unor9,unor7,ac,whf,wh1, tf140m,
glf,glf1, whc,wh3,tf141m,glc,glc1,tf147,tf148,tf149,
tf150,tf151,tf152,tf149m,tf150m)
```

```
if tf150>= 0 then
tf153=tf150/5

if tf149>=0 then
tf154=tf149/5

if tf153 > 999999.9 then
tf153=999999.9

if tf154 > 999999.9 then
tf154=999999.9
run tf155(tf156, tf157, tf158, tf159,tf160,tf161,tf162,tf163,
f164, f165,tf166,tf167,tf168,tf169,tf170,tf171, tf172, tf173,
tf174, tf68,tf69,no3,unor8,ac,whr,wh2,tf174m,glr,glr1, tf175,
tf176, tf177,tf176m)

if tf176>= 0 then
tf178=tf176/5


if tf178 > 999999.9 then
tf178=999999.9
run tf179(tf180,tf1810,tf182,tf183,tf184,tf185, tf186, tf187,
tf188,tf189,tf190,tf191,tf192,no3,unor6,tf193,tf194)

run tf195(tf196,tf197,tf198,tf199,tf200,tf201,tf202,tf203,
tf204, tf205, tf206,tf207,tf208,tf209,tf210,tf211,tf212,
tf213, tf214,tf215,no3,unor6,tf216,tf217)

run tf218(tf219,tf220,tf221,tf222,tf223, tf224, tf225, tf226,
tf227,tf228,tf229,tf230,tf231,tf232,tf233,tf234,tf235, tf236,
no3,unor6,tf237,tf238)

if tf216>=0  then
tf239=tf216+tf237;
tf240=tf239+tf193;
dddd=tf242;
tf241=tf240/dddd;
tf243=tf241/8;
tf244=tf243/5;
tf245=tf194+tf217+tf238;

if ac = 1 then
tf243m=tf243*40/whf+wh1

if ac =2 and tf242m>0 then
ffm=tf243*tf242/tf242m;
run Aa(tf242,tf242m,ffm,glf,glf1,tf243m,af)

if tf243 > 999999.9 then
tf243=999999.9

if tf243m > 999999.9 then
tf243m=999999.9

if tf239 > 999999.9 then
tf239=999999.9
```

```
if tf240 > 999999.9 then
tf240=999999.9

if tf241 > 999999.9 then
tf241=999999.9

if tf245 > 999999.9 then
tf245=999999.9

if tf244 > 999999.9 then
tf244=999999.9

runtf31(tf246,tf247,tf248,tf249,tf250,tf251,tf252,tf253,tf254
,tf255,tf256,tf257,tf258,tf259,tf260,tf261,tf262,tf263,tf264,
tf265,tf282,tf68,no3,unor5,ac,whr,whlr,tf282m,glr,glr1,tf285,
tf286,tf287,tf286m,wh2)

if tf286>=0 then
tf288=tf286/5

if tf288 > 999999.9 then
tf288=999999.9

run tf289(tf122,tf123,tf124,tf125,tf126, tf127, tf128, tf129,
tf130, tf131,tf132,tf290,tf291,tf292,tf293,tf294,tf295,tf296,
tf297,tf298,tf299,tf300,tf301,no3,unor4,ac, whc, wh3, tf290m,
glc,glc1,tf302,tf303,tf304,tf303m)

if tf303>=0 then
tf305=tf303/5

if tf305 > 999999.9 then
tf305=999999.9

run time_total3(tf27,tf147,tf241,tf70,tf175,tf285,tf110,
tf148, tf302, tf307)


run tf306(tf28,tf71,tf111, tf149, tf150, tf176, tf243, tf286,
tf303,tf308)

run tf306(tf30,tf73,tf134, tf154, tf153, tf178, tf244, tf288,
tf305, tf309)

run tf310(tf8,tf9,tf10,tf11,tf12,tf13,tf14, tf15, tf16, tf32,
tf33, tf34,tf35,tf36,tf37,tf38,tf39,tf40,tf41,tf42,tf43,tf44,
tf45, tf46,tf47,tf48,tf49,tf50,tf51,tf312,tf313,tf314, tf315,
tf316,tf317,tf318,tf319,tf320,tf321,tf322,tf323,tf324, tf325,
tf326,tf327,tf328,tf329,tf330,tf321,tf322,tf323,tf324, tf325,
tf326,tf327,tf328,tf329,tf330,no3,tf339,tf340)

run tf341(tf136,tf137,tf138,tf139,tf156, tf157, tf158, tf159,
tf160, tf161,tf162,tf163,tf164,tf165,tf166,tf167,tf168,tf169,
tf170,tf171,tf172,tf173tf342,tf343,tf344,tf345, tf321, tf322,
tf323,tf324,tf325,tf326,tf327,tf328,tf329,tf330,tf331, tf332,
tf333,tf334,tf335,tf336,tf337,tf338,no3,tf346,tf347)
```

```
run tf348(tf180,tf1810,tf182,tf183,tf184,tf185, tf186, tf187,
tf196,tf197,tf198,tf199,tf200,tf201,tf202,tf203,tf204, tf205,
tf206,tf207,tf219,tf220,tf221,tf222,tf223,tf224,tf225, tf226,
tf227,tf228,tf229,tf349,tf350,tf351,tf349,tf350,tf351, tf352,
tf353,tf354,tf355,tf356,tf357,tf354,tf355,tf356,tf357, tf358,
tf359,tf360,tf361,tf362,tf363,tf364,tf365,tf362,tf363, tf364,
tf365,tf366,tf367,tf368,no3,tf369)

run tf370(tf246,tf247,tf248,tf249,tf250, tf251, tf252, tf253,
tf254, tf255, tf256, tf257, tf258,tf259, tf260, tf261, tf262,
tf263, tf264, tf265, tf266,tf267, tf268, tf269, tf270, tf271,
tf272,tf273,tf274,tf275,tf276,tf277,tf278,tf279,tf280, tf281,
tf321,tf322,tf323,tf324,tf325,tf326,tf327,tf328,tf329, tf330,
tf321,tf322,tf323,tf324,tf325,tf326,tf327,tf328,tf329, tf330,
tf331,tf332,tf333,tf334,tf335,tf336,tf337,tf338,tf331, tf332,
tf333,tf334,tf335,tf336,tf337,tf338,no3,tf371)

run tf372(tf29,tf245,tf151, tf72, tf287, tf177, tf112, tf304,
tf152,tf339,tf369,tf346,tf340,tf371,tf347,tf116,tf133, tf120,
wh1, wh2,wh3,tf17,tf242,tf140,tf67,tf282, tf174, tf91, tf290,
tf141, tf17m,tf242m,tf140m,tf67m,tf282m,tf174m,tf91m, tf290m,
tf141m, glf,glf1,glr,glr1,glc, glc1, ac, tf373, tf374, tf375,
tf376, tf377, tf378,tf379,tf380,tf381,tf373m, tf374m, tf375m,
tf376m, tf377m, tf378m,tf379m,tf380m,tf381m)

run fltco(tf373,tf374,tf375,tf376,tf377, tf378, tf379, tf380,
tf381,fltcos)

run fltco(tf373m,tf374m,tf375m,tf376m,tf377m, tf378m, tf379m,
tf380m,tf381m,fltcosm)

if  lag_cw1>=0 and lag_cw2>=0 or lag_sb1>=0 or lag_sb2>0 or
lag_sc1>=0 or lag_sc2>=0 or lag_cwsb1>=0 or lag_sbs1>=0 then
run ftt(tf28,tf71,tf111,tf243,tf286,tf303,tf149,tf176, tf150,
lag_cw1,lag_cw2,lag_sb1,lag_sb2,lag_sc1,lag_sc2,lag_cwsb1,
lag_sbs1,wallday,slabday,stairday,fdat,fhrt,fwet)

if lag_cw1m>=0 and lag_cw2m>=0 or lag_sb1m>=0 or lag_sb2m>0
or lag_sc1m>=0 or lag_sc2m>=0 or lag_cwsb1m>=0 or
lag_sbs1m>=0 then

run ftt(tf28m, tf71m, tf111m, tf243m, tf286m, tf303m, tf149m,
tf176m,tf150m,lag_cw1m,lag_cw2m,lag_sb1m,lag_sb2m,lag_sc1m,
lag_sc2m,lag_cwsb1m,lag_sbs1m,walldaym,slabdaym,stairdaym,
fdatm,fhrtm,fwetm)

run con_tyf(tf8,tf9,tf10,tf11,tf12,tf13,tf14,tf15,tf16, tf32,
tf33,tf34,tf35,tf36,tf37,tf38,tf39,tf40,tf41,tf42,tf43, tf44,
tf45,tf46,tf47,tf48,tf49,tf50,tf51,tf52,tf53,tf54,tf55, tf56,
tf57,tf58,tf59,tf60,tf61,tf62,tf63,tf64,tf65,tf66,tf75, tf76,
tf77,tf78,tf79,tf80,tf81,tf82,tf83,tf84,tf85,tf86,tf87, tf88,
tf89,tf90,tf136,tf137,tf138,tf139,tf118, tf119, tf156, tf157,
tf158,tf159,tf160,tf161,tf162,tf163,tf164,tf165,tf166, tf167,
tf168,tf169,tf170,tf171,tf172,tf173,tf180,tf1810,tf182,tf183,
tf184,tf185,tf186,tf187,tf311, typ)

run con_tyfa(tf17,tf67,tf91,tf140,tf174, tf242, tf282, tf290,
tf141,tf196,tf197,tf198,tf199,tf200,tf201,tf202,tf203, tf204,
```

```
tf205,tf206,tf207,tf219,tf220,tf221,tf222,tf223,tf224, tf225,
tf226,tf227,tf228,tf229,tf246,tf247,tf248,tf249,tf250, tf251,
tf252,tf253,tf254,tf255,tf256,tf257,tf258,tf259,tf260, tf261,
tf262,tf263,tf264,tf265,tf266,tf267,tf268,tf269,tf270, tf271,
tf272,tf273,tf274,tf275,tf276,tf277,tf278,tf279,tf280, tf281,
tf122,tf123,tf124,tf125,tf126,tf127,tf128,tf129,tf130,tf131,
tf132,typa)

run cost_typ(tf18,tf19,tf20,tf21,tf22,tf23, tf24, tf25, tf26,
tf68,tf69,tf92,tf93,tf94,tf95, tf96, tf97, tf98, tf99, tf100,
tf101,tf102,tf103,tf105,tf106,tf107,tf114,tf115,tf142, tf143,
tf144,tf145,tf146,tf188,tf189,tf190,tf191,tf192,tf208, tf209,
tf210,tf211,tf212,tf213,tf214,tf215,tf230,tf231,tf232, tf233,
tf234,tf235,tf236,tf291,tf292,tf293,tf294,tf295,tf296, tf297,
tf298,tf299,tf300,tf301,tf312,tf313,tf314,tf315,tf316,fltcos,
ctyp)

run cost_typa(tf354,tf355, tf356, tf357, tf358, tf359, tf360,
tf361,tf362,tf363,tf364,tf365,tf366,tf367,tf368,tf317, tf318,
tf319,tf320,tf321,tf322,tf323,tf324,tf325,tf326,tf327, tf328,
tf329,tf330,tf331,tf332,tf333,tf334,tf335,tf336,tf337, tf338,
tf342,tf343,tf344,tf345,tf346,tf347,tf349,tf350,tf351,tf352,
tf353,tf104,ctypa)

run num3(no3,noo3)

run gratyf(tf30,tf244,tf154,tf73, tf288, tf178, tf134, tf305,
tf153,grtyf)

run gratyf2(lag_cw1,lag_cw2,lag_sb1,lag_sb2,lag_sc1, lag_sc2,
lag_cwsb1,lag_sbs1,grtyf2)

run nstore(new_project,newww)
if st3 is '(5)' then
use screen_final3

if st3 is '(4)' or
st3 is '(6)' then
use screen_optf1

run co_opt(fltcos,fdat,tf373,tf374,tf375,tf376, tf377, tf378,
tf379,tf380,tf381,tf28,tf149,tf243,tf71, tf176, tf286, tf111,
tf150,tf303,tf373m,tf374m,tf375m,tf376m,tf377m,tf378m,tf379m,
tf380m,tf381m,tf28m,tf149m,tf243m,tf71m,tf176m,tf286m,tf111m,
tf150m,tf303m,fltcosm,fdatm,tf17m,tf140m,tf242m,tf67m,tf174m,
tf282m,tf91m,tf141m,tf290m,fltcoos)

run co_optf( wallday, slabday, stairday, walldaym, slabdaym,
stairdaym,fdat,fdatm,fltcos,fltcosm,tf381,tf381m,flopt)


runora ( unor1, unor2, unor3, unor4, unor5,unor6,unor7,unor8,
unor9,whf,whr,whc,whlf,whlr,whlc,glf,glr,glc,glf1,glr1, glc1,
unor)

if st3 is not '(8)' and
lag_cw1>=0 and
lag_cw2>=0 and
```

```
lag_sb1>=0 and
lag_sb2>=0 and
lag_sc1>=0 and
lag_sc2>=0 and
lag_cwsb1>=0 and
lag_sbs1>=0 then
run floor_return()

run dummy2(new_project)

run nstore1(new_project,new1)

run ty_escape()
```

'ENVELOPE'

```
seek enn

use screen_en

run enc(en,en1)

run esc()
```

'EXTERNAL BRICKWORK/BLOCKWORK'

```
seek block_brick

control common

run con_bbo2(bbof)

run con_bbof2(mur)

run cost_bb2(cbbof)

run co_opt2(bctoot)

run noooob3(nob3)

run ntore2(new2)

if new2 is 'y'   then
run conbbo(b6,b7,b8,b9,b10,b11,b12,b13,b14,b15,b27, b28, b29,
b30,b31,b32,b33,b34,b44,b45,b46,b47,b48,b49,b50,b51,b52, b53,
b54,b55,b56,b57,b58,b59,b60,b61,b76,b77,b78,b79,b80,b81, b82,
b83,b84,b85,b86,b87,b88,b89,b90,b91,b92, b93, b94, b95, b111,
b112,b113,b114,b115,b116,b117,b118,b119,b120,b136,b137, b138,
b139,b140,b141,b142,b143,b144,b145,b159,b167,nb3,b159m,b167m,
b217m,b219m,b161m,b169m,bf136,bf137,bf144,bf207,bf146,bf211,
bf150,bf209,bf147,bctotm);
b173m=b161m

use screen_4st
```

```
if st4 is '(1)' then
use screen_41;
ac=3

if st4 is '(2)' then
use screen_41c;
ac=3

if st4 is '(3)' then
use screen_gangsize4;
ac=3

if st4 is '(5)' then
use screen_ac

if st4 is '(6)' then
use screen_oub;
ac=3

if st4 is '(7)' then
ac=3;
run st_return

if st4 is '(4)' then
ac=3

if ac=1  then
use screen_ov;

if ac=2  then
use screen_gangsize4;
use screen_lab

if ac <> 1 and
ac<> 2 then
ac=3

run b5(b6,b7,b8,b9,b10,b11,b12,b13,b14,b15,b16,b17, b18, b19,
b20, b21, ebloc,b22,b23,b24,b25)

run b26(b27,b28,b29,b30,b31,b32,b33, b34, b35, b36, b37, b38,
b21,ebloc,b39,b40,b41,b42)

run b43(b44,b45,b46,b47,b48,b49,b50,b51,b52,b53,b54,b55, b56,
b57,b58,b59,b60,b61,b62,b63,b64,b65,b66,b67,b68,b69,b70, b21,
ebloc,b71,b72,b73,b74)

run b75(b76,b77,b78,b79,b80,b81,b82,b83,b84,b85,b86, b87,b88,
b89, b90,b91,b92,b93,b94,b95,b96,b97,b98,b99,b100, b101,b102,
b103,b104,b105,b21,ebloc,b106,b107,b108,b109)


run b110(b111,b112,b113,b114,b115,b116,b117,b118, b119, b120,
b121,b122,b123,b124,b125,b126,b127,b128,b129,b130,ebloc,b131,
b132,b133,b134)

run b135(b136,b137,b138,b139,b140,b141,b142,b143, b144, b145,
b146,b147,b148,b149,b150,b151,b152,bf136,bf137,bf144,bf146,
```

```
bf147,bf150,ebric, b153,b154,b155,b156)

if b131 >=0 then
b157=b22+b39+b71;
b158=b157+b131

if b159=0 then
b160=0;
b161=0;
b162=0
if b159m=0 then
b161m=0

if b159>0 then
nb3160=nb3*b159;
b160=b158/nb3160;
b161=b160/8;
b165=b24+b41+b73;
b166=b165+b108+b133;
b162=b166/nb3160

if ac=1 and b159>0 then
wh1=whl1*whbl/100-0.40*whl1;
b161m=b158/b159/8/nb3*40/whbl+wh1

if ac=2 then
ffm1=b158/b159m/8/nb3;
run Aa(b159m,b159,ffm1,glf1,gll1,b161m,ra1)

if b167=0 then
b168=0;
b169=0;
b170=0

if b167m=0 then
b169m=0

if b167>0 then
nb3168=nb3*b167;
b168=b153/nb3168;
b169=b154/nb3168;
b170=b155/nb3168

if ac=1 and b167>0 then
wh2=whlr*whbr/100-0.40*whlr;
b169m=b153/b167/8/nb3*40/whbr+wh2

if ac=2 then
ffm2=b153/b167m/8/nb3;
run Aa(b167m,b167,ffm2,glfr,glr1,b169m,ra2)

if b160<b168 then
b171=b168;
b172=b169;
b173=b170

if b160>=b168 then
b171=b160;
```

```
b172=b161;
b173=b162

if b161m>b169m then
b173m=b161m

if b161m<=b169m then
b173m=b169m

if b171 > 9999.9 then
b171=9999.9

if b160> 9999.9 then
b160=9999.9

if b161> 9999.9 then
b161=9999.9

if b162> 9999.9 then
b162=9999.9

if b168> 9999.9 then
b168=9999.9

if b169> 9999.9 then
b169=9999.9

if b170> 9999.9 then
b170=9999.9

if b161m> 9999.9 then
b161m=9999.9

if b169m> 9999.9 then
b169m=9999.9

if b172> 99999.9 then
b172=99999.9

if b173m> 99999.9 then
b173m=99999.9

run b174(b6,b7,b8,b9,b10,b11,b12,b13,b14,b15,b27,b28,b29,b30,
b31,b32,b33,b34,b44,b45,b46,b47,b48,b49,b50,b51,b52,b53,b175,
b176,b177,b178,b179,b180,b181,b182,b183,b184,b185,b186 ,b187,
b188,b189,b190)

run b174(b54,b55,b56,b57,b58,b59,b60,b61,b76,b77,b78,b79,b80,
b81,b82,b83,b84,b85,b86,b87,b88,b89,b90,b91,b92,b93,b94, b95,
b191,b192,b193,b194,b195,b196,b197,b198,b199,b200,b201, b202,
b203,b204,b189,b205)

run b206(b136,b137,b138,b139,b140,b141,b142,b143, b144, b145,
b207,b208,b209,b210,b211,b212,b152,bf136,bf207, bf144, bf211,
bf137,bf209,b213)

run b214(b190,b205,b215)
```

```
'INTERNAL BRICKWORK/BLOCKWORK'

seek block_brick

control common

run con_bbo2(ibbof)

run con_bbof2(imur)

run cost_bb2(icbbof)

run co_opt2(ibctoot)

run noooob3(nob3)

run ntore2(new2)

if new2 is 'y'   then
run conbbo(ib6,ib7,ib8,ib9,ib10,ib11, ib12, ib13, ib14, ib15,
ib27, ib28,ib29,ib30,ib31,ib32,ib33,ib34,ib44,ib45,ib46,ib47,
ib48,ib49,ib50,ib51,ib52,ib53,ib54,ib55,ib56,ib57,ib58, ib59,
ib60,ib61,ib76,ib77,ib78,ib79,ib80,ib81,ib82,ib83,ib84, ib85,
ib86,ib87,ib88,ib89,ib90,ib91, ib92, ib93, ib94, ib95, ib111,
ib112,ib113,ib114,ib115,ib116,ib117,ib118,ib119,ib120, ib136,
ib137,ib138,ib139,ib140,ib141,ib142,ib143,ib144,ib145, ib159,
ib167,nb3,ib159m,ib167m,ib217m,ib219m,ib161m, ib169m, ibf136,
ibf137,ibf144,ibf207, ibf146, ibf211, ibf150, ibf209, ibf147,
ibctotm);
ib173m=ib161m

use screen_4st

if st4 is '(1)' then
use screen_41;
ac=3

if st4 is '(2)' then
use screen_41c;
ac=3

if st4 is '(3)' then
use screen_gangsize4;
ac=3

if st4 is '(5)' then
use screen_ac;

if st4 is '(6)' then
use screen_oub;
ac=3

if st4 is '(7)' then
ac=3;
run st_return()

if ac =1 then
use screen_ov
```

```
if ac=2 then
use screen_gangsize4;
use screen_lab

if st4 is '(4)' then
ac=3

run b5(ib6,ib7,ib8,ib9,ib10,ib11,ib12,ib13, ib14, ib15, ib16,
ib17, ib18,ib19,ib20,ib21,ibloc,ib22,ib23,ib24,ib25)

run b26 (ib27,ib28,ib29,ib30,ib31,ib32,ib33,ib34, ib35, ib36,
ib37,ib38,ib21,ibloc,ib39,ib40,ib41,ib42)

run b43 (ib44,ib45,ib46,ib47,ib48,ib49,ib50,ib51, ib52, ib53,
ib54,ib55,ib56,ib57,ib58, ib59, ib60, ib61, ib62, ib63, ib64,
ib65, ib66,ib67,ib68,ib69,ib70,ib21,ib71,ib72,ib73,ib74)

run  b75(ib76,ib77,ib78,ib79,ib80,ib81,ib82,ib83, ib84, ib85,
ib86,ib87,ib88,ib89,ib90,ib91,ib92,ib93,ib94,ib95,ib96, ib97,
ib98,ib99,ib100,ib101,ib102,ib103, ib104, ib105, ib21, ib106,
ib107,ib108,ib109)

run b110(ib111,ib112,ib113,ib114, ib115, ib116, ib117, ib118,
ib119,ib120,ib121,ib122,ib123,ib124,ib125,ib126,ib127, ib128,
ib129,ib130,ibloc,ib131,ib132,ib133,ib134)

run b135(ib136,ib137,ib138,ib139, ib140, ib141, ib142, ib143,
ib144,ib145,ib146,ib147,ib148,ib149,ib150,ib151,ib152,ibf136,
ibf137,ibf144,ibf146,ibf147,ibf150,ibric,ib153, ib154, ib155,
ib156)



if ib131 >=0 then
ib157=ib22+ib39+ib71;
ib158=ib157+ib131

if ib159=0 then
ib160=0;
ib161=0;
ib162=0

if ib159m=0 then
ib161m=0

if ib159>0 then
inb3160=nb3*ib159;
ib160=ib158/inb3160;
ib161=ib160/8;
ib165=ib24+ib41+ib73;
ib166=ib165+ib108+ib133;
ib162=ib166/inb3160

if ac=1 and ib159>0 then
wh1=wh11*whb1/100-0.4*wh11;
ib161m=ib158/ib159/8/nb3*40/whb1+wh1

if ac =2 then
```

```
ffm1=ib158/ib159m/8/nb3;
run Aa(ib159m,ib159,ffm1,glf1,gll1,ib161m,ra1)

if ib167=0 then
ib168=0;
ib169=0;
ib170=0

if ib167m=0 then
ib169m=0

if ib167>0 then
inb3168=nb3*ib167;
ib168=ib153/inb3168;
ib169=ib154/inb3168;
ib170=ib155/inb3168

if ac=1 and ib167>0 then
wh2=whlr*whbr/100-0.4*whlr;
ib169m=ib153/ib167/8/nb3*40/whbr+wh2

if ac=2 then
ffm2=ib153/ib167m/8/nb3;
run Aa(ib167m,ib167,ffm2,glfr,glr1,ib169m,ra2)

if ib160<ib168 then
ib171=ib168;
ib172=ib169;
ib173=ib170

if ib160>=ib168 then
ib171=ib160;
ib172=ib161;
ib173=ib162

if ib161m>ib169m then
ib173m=ib161m

if ib161m<=ib169m then
ib173m=ib169m


if ib171 > 9999.9 then
ib171=9999.9

if ib160> 9999.9 then
ib160=9999.9

if ib161> 9999.9 then
ib161=9999.9

if ib162> 9999.9 then
ib162=9999.9

if ib168> 9999.9 then
ib168=9999.9

if ib169> 9999.9 then
```

```
ib169=9999.9

if ib170> 9999.9 then
ib170=9999.9

if ib161m> 9999.9 then
ib161m=9999.9

if ib169m> 9999.9 then
ib169m=9999.9

if ib172> 99999.9 then
ib172=99999.9

if ib173m> 99999.9 then
ib173m=99999.9
runb174(ib6,ib7,ib8,ib9,ib10,ib11,ib12,ib13,ib14, ib15, ib27,
ib28,ib29,ib30,ib31,ib32,ib33,ib34,ib44,ib45,ib46,ib47, ib48,
ib49,ib50,ib51,ib52,ib53,ib175,ib176,ib177,ib178,ib179,ib180,
ib181,ib182,ib183,ib184,ib185,ib186,ib187, ib188, ib189, nb3,
ib190)

run b174(ib54,ib55,ib56,ib57,ib58,ib59,ib60,ib61, ib76, ib77,
ib78,ib79,ib80,ib81,ib82, ib83, ib84, ib85, ib86, ib87, ib88,
ib89,ib90,ib91,ib92,ib93,ib94,ib95,ib191,ib192, ib193, ib194,
ib195,ib196,ib197,ib198,ib199,ib200,ib201,ib202, ib203,ib204,
ib189, nb3,ib205)

run b206(ib136,ib137,ib138,ib139, ib140, ib141, ib142, ib143,
ib144,ib145,ib207,ib208,ib209,ib210,ib211,ib212,ib152,ibf136,
ibf207, ibf144,ibf211,ibf137,ibf209,ib213)


run b214(ib190,ib205,ib215)

run b216(ib215,ib25,ib42,ib74,ib109,ib134,nb3, ac,wh1,ib159m,
ib159,glfl,gll1,ib217,ib217m)

run b218(ib213,ib156,nb3,ac,wh2,ib167m,ib167,glfr,glr1,ib219,
ib219m)

run bcctot(ib217,ib219,ibctot)

run bcctot(ib217m,ib219m,ibctotm)

run con_bbo(ib6,ib7,ib8,ib9,ib10,ib11,ib12, ib13, ib14, ib15,
ib27, ib28,ib29,ib30,ib31,ib32,ib33,ib34,ib44,ib45,ib46,ib47,
ib48, ib49,ib50,ib51, ib52,ib53,ib54,ib55,ib56,ib57,ib58,ib59
,ib60, ib61, ib76, ib77,ib78,ib79,ib80,ib81,ib82, ib83, ib84,
ib85,ib86,ib87,ib88,ib89,ib90,ib91,ib92,ib93,ib94,ib95,ib111,
ib112,ib113,ib114,ib115,ib116,ib117,ib118,ib119,ib120, ib136,
ib137, ib138,ib139,ib140,ib141,ib142,ib143,ib144,ib145,ib159,
ib167, ibbof)

run cost_bb(ib16,ib17,ib18,ib19,ib20, ib21, ib35, ib36, ib37,
ib38,ib62,ib63,ib64,ib65,ib66,ib67,ib68,ib69,ib70,ib96, ib97,
ib98, ib99,ib100,ib101,ib102,ib103,ib104,ib105, ib121, ib122,
ib123, ib124,ib125,ib126,ib127,ib128,ib129,ib130,ib146,ib147,
```

```
ib148, ib149,ib150, ib151, ib152, ib175, ib176, ib177, ib178,
ib179, ib180, ib181, ib182,   ib183, ib184,ib185,ib186,ib187,
ib188,ib189,ib191,ib192,ib193,ib194,ib195,ib196,ib197, ib198,
ib199,ib200, ib201,ib202,ib203,ib204,ib207,ib208,ib209,ib210,
ib211, ib212,icbbof)

run btt(ib160,ib168,ib161,ib169,ib162,ib170,ibhrt,ibwet)

run co_opt(ibctot,ib172,ibctotm,ib173m,ib159m, ib167m, ib217,
ib219,ib161, ib169,ib217m,ib219m,ib161m,ib169m, ibloc, ibric,
whbr, whbl, whlr,whll,glfr,glfl,glrl,glll,ibctoot)

run conbbof (ibf136, ibf137, ibf144, ibf207, ibf146, ibf211,
ibf150, ibf209, ibf147, imur)

run grabb(ib162,ib170,grabbbi)

run dummy2 (newi22)

if st4 is '(1)' or
st4 is '(2)' or
st4 is '(3)' then
run bbloc_return()


if st4 is '(5)' then
use screen_optbb;
run bbloc_return()

if st4 is '(4)' then
use screen_final4;
run bbloc_return()

run dummy2(new_project)

run nstore2(new_project,new2)

run bb_escape()
```

'FINISHES'

```
seek finis

run fin2(fif)

use screen_f1

run fin(fii2,fiw2,fif2,fic2,fie2,fiil,fiwl, fifl, ficl, fiel,
fii2m,fiw2m, fif2m, fic2m, fie2m, fiilm, fiwlm, fiflm, ficlm,
fielm, fif)

run st_return

run fin_esc()
```

```
'SERVICES'

seek servs

run fin2(ser)

use screen_f1

run fin(sel2,sem2,see2,sel1, sem1, see1, sel2m, sem2m, see2m,
sel1m,sem1m, see1m,ser)


run st_return()

run fin_esc()


'ROOF ASPHALT'

seek roofa

run ra2(raa)

use screen_r1

use screen_ac

if ac =1 then
glfr=0;
glr1=0;
use screen_ov

if ac=2 then
whbr=0;
whlr=0;
use screen_lab

run cot(laq1,laq2,laq3,ulc1,ulc2,ulc3,umc1,umc2,umc3,gs,rnor,
ac,whbr, whlr,glfr,glr1,gsm,costra,durra,costram,durram)

if durra>=0 then
dayra=durra/8;
weekra=dayra/5

if durra>=999999.9 then
durra=999999.9

if dayra>=999999.9 then
dayra=999999.9

if weekra>=999999.9 then
weekra=999999.9

if costra>=999999.9 then
costra=999999.9

if durram>=0 then
```

```
dayram=durram/8;
weekram=dayram/5

if durram>=999999.9 then
durram=999999.9

if dayram>=999999.9 then
dayram=999999.9

if weekram>=999999.9 then
weekram=999999.9

if costram>=999999.9 then
costram=999999.9

use screen_finalra

run ra(costra,dayra,costram,dayram,weekra, durra, laq1, laq2,
laq3,ulc1,ulc2,ulc3,umc1,umc2,umc3,gs, rnor, gsm, whbr, whlr,
glfr,glr1, raa)

run ragr(weekra,ragrf)
run ra_escape()


'INDIRECT COSTS'

seek ic

run coo2(iic)

use screen_ic

run co(ic1,ic2,ic3,ic4,ic5,ic6,ic7,ic8,ic9,icc)

run coo(icc,ic1,ic2,ic3,ic4,ic5,ic6,ic7,ic8,ic9,iic)

run st_return()

run es_c()


LAG VALUES'


seek main_lag

run lag_mains2(lagmain)

run lag_mains2m(lagmainm)

use screen_d

use screen_exc_lap

use screen_ty_lap
```

```
run num3a(noo3)

run coo2(iic)

if lt_lr >=0 or
lr_lb>=0 or
lr_lb1>=0 or
lag_sifo>=0 or
f1_f2>=0 or
f2_f3>=0 or
lag_laro>=0 or
lag_cw2>=0 or
lag_cw1>=0 or
lag_sb1>=0 or
lag_sb2>=0 or
lag_sc1>=0 or
lb_lt>=0 or
lt_lc>=0 or
lag_sbs1>=0 or
b173m>=0 or
tf381>0 or
dayra>=0 or
lr_lb1>=0 or
lag_fogr>=0 or
lag_grfir>=0 or
lag_ra>=0 or
lag_cwsb1>=0 or
no3>=0 then
run dur1(lt_lr,lr_lb,lr_lb1,lag_sifo, f1_f2, f2_f3, lag_fogr,
lag_grfir,lag_cwsb1,lag_cw2,lag_cw1,lag_sb1, lag_sb2, lag_ra,
dayra, no3, day1)

if lt_lrm >=0 or
lr_lbm>=0 or
lr_lb1m>=0 or
lag_sifom>=0 or
f1_f2m>=0 or
f2_f3m>=0 or
lag_larom>=0 or
lag_sc1m>=0 or
lb_ltm>=0 or
lt_lcm>=0 or
lag_sbs1m>=0 or
lr_lb1m>=0 or
lag_fogrm>=0 or
lag_grfirm>=0 or
lag_ram>=0 or
lag_cwsb1m>=0 or
lag_sc2m >=0 then
emel=0

if sdat>=0 or
walldaym >=0 or
lag_cwsb1>=0 or
wallday >=0 or
slabday>=0 or
stairday>=0 or
```

```
run num3a(noo3)

run coo2(iic)

if lt_lr >=0 or
lr_lb>=0 or
lr_lb1>=0 or
lag_sifo>=0 or
f1_f2>=0 or
f2_f3>=0 or
lag_laro>=0 or
lag_cw2>=0 or
lag_cw1>=0 or
lag_sb1>=0 or
lag_sb2>=0 or
lag_sc1>=0 or
lb_lt>=0 or
lt_lc>=0 or
lag_sbs1>=0 or
b173m>=0 or
tf381>0 or
dayra>=0 or
lr_lb1>=0 or
lag_fogr>=0 or
lag_grfir>=0 or
lag_ra>=0 or
lag_cwsb1>=0 or
no3>=0 then
run dur1(lt_lr,lr_lb,lr_lb1,lag_sifo, f1_f2, f2_f3, lag_fogr,
lag_grfir,lag_cwsb1,lag_cw2,lag_cw1,lag_sb1, lag_sb2, lag_ra,
dayra, no3, day1)

if lt_lrm >=0 or
lr_lbm>=0 or
lr_lb1m>=0 or
lag_sifom>=0 or
f1_f2m>=0 or
f2_f3m>=0 or
lag_larom>=0 or
lag_sc1m>=0 or
lb_ltm>=0 or
lt_lcm>=0 or
lag_sbs1m>=0 or
lr_lb1m>=0 or
lag_fogrm>=0 or
lag_grfirm>=0 or
lag_ram>=0 or
lag_cwsb1m>=0 or
lag_sc2m >=0 then
emel=0

if sdat>=0 or
walldaym >=0 or
lag_cwsb1>=0 or
wallday >=0 or
slabday>=0 or
stairday>=0 or
```

```
laf1m>=0 or
lacem>=0 or
laexm>=0 or
lalim>=0 or
lamem>=0 or
laelm>=0 or
fii2m>=0 or
fiw2m>=0 or
fif2m>=0 or
fic2m>=0 or
fie2m>=0 or
sel2m>=0 or
sem2m>=0 or
see2m>=0 then
run dur3(lt_lrm, lr_lbm, lr_lb1m, lag_sifom, f1_f2m, f2_f3m,
lag_fogrm, lag_grfirm,lag_cwsb1m,lag_nfbbm,nofm,lainm, lawam,
laf1m, lacem,laexm,lalim,lamem,laelm,fii2m,fiw2m,fif2m,fic2m,
fie2m,sel2m,sem2m,see2m, lag_cw2m,lag_cw1m,lag_sb1m,lag_sb2m,
day3m, d1m,d2m,d3m,d4m,d5m,d6m,d7m,d8m)

run tot_dur(day1,day2,day3,day1m,day2m,day3m,day,daym)

if fii1>=0 or
fiw1>=0 or
fif1>=0 or
fic1>=0 or
fie1>=0 or
sel1>=0 or
sem1>=0 or
see1>=0 or
sitotc>=0 or
cofotot>=0 or
costra>=0 or
fltcos>=0 or
ibctot>=0 or
bctot>=0 then
run tot_cost(fii1,fiw1,fif1,fic1,fie1,sel1,sem1,see1, sitotc,
cofotot,costra,no3,fltcos,ibctot,bctot,dir)

if fii1m>=0 or
fiw1m>=0 or
fif1m>=0 or
fic1m>=0 or
fie1m>=0 or
sel1m>=0 or
sem1m>=0 or
see1m>=0 or
sitotcm>=0 or
cofototm>=0 or
costram>=0 or
fltcosm>=0 or
ibctotm>=0 or
bctotm>=0 then

run tot_cost(fii1m,fiw1m,fif1m,fic1m,fie1m,sel1m,sem1m,see1m,
sitotcm, cofototm,costram,no3,fltcosm,ibctotm,bctotm,dirm)
```

```
if icc>=0 then
run ind_tot(icc,day,daym,dir,dirm,iccm,tot,totm)

use screen_tot

run co_opt (day,daym,dir,dirm,icc,iccm,tot,totm,day1,day2,
day3, day1m, day2m,day3m,d1,d2,d3,d4,d5,d6,d7,d8,d1m,d2m,
d3m, d4m, d5m, d6m, d7m,d8m, tott)

run tot_es
```

```
DECLARE SUB p1p12g (gxxe(), gxx1!(), gyye!(), gyy1!(),
gzze!(), gzz1!(), i!, a!)
DECLARE SUB p2p22 (gxxee(), gxxeei!(), nof!, nof1!, page!,
qxxe!(), qxx!(), qyye!(), qyy!(), qzze!(), qzz!(), i!, a!)
DECLARE SUB p1p10 (g1xxe, g1xx1!, g1yye!, g1yy1!, g1zze!,
g1zz1!, g1tte!, g1tt1!, g1uue!, g1uu1!, g22b!, g22b1!,
g2xxe!, g2xx1!, g2yye!, g2yy1!, g2zze!, g2zz1!, gyye!(),
gyy1!(), gxxe!(), gxx1!())
DECLARE SUB p1p10e (xxe, xx!, yye!, YY!, zze!, zz!, tte!,
tt!, uue!, uu!, xxxe2!, xxx2!, xxe2!, xx2!, yye2!, yy2!,
zze2!, zz2!, qyye!(), qyy1!(), qxxe!(), qxx!())
DECLARE SUB ngigi (fin1, FIN2!, fin3!, fin4!, fin5!, fin6!,
fin7!, fin8!, fin9!, fin10!, ser1!, ser2!, ser3!, ser4!,
ser5!, ser6!, afin1!(), afin2!(), afin3!(), afin4!(),
afin5!(), afin6!(), afin7!(), afin8!(), afin9!(), afin10!(),
aser1!(), aser2!(), aser3!(), aser4!(), aser5!(), aser6!(),
t!, jf1!, jf3!, jf5!, jf7!, jf9!, js1!, js3!, js5!, gig!,
nofloor!)
DECLARE SUB keyin (MINKEY(), MAXKEY!(), EXTMINKEY!(),
EXTMAXKEY!(), SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON!(),
CALLER!, XFLD!, YFLD!, LNGTH!, TRUE!, FALSE!, c$)
DECLARE SUB sf (afin1(), afin2!(), afin3!(), afin4!(),
afin5!(), afin6!(), afin7!(), afin8!(), afin9!(), afin10!(),
aser1!(), aser2!(), aser3!(), aser4!(), aser5!(), aser6!(),
i!)
DECLARE SUB sff (gxxee(), gxx1e!(), fin1!, FIN2!, fin3!,
fin4!, fin5!, fin6!, fin7!, fin8!, fin9!, fin10!, ser1!,
ser2!, aser3!, aser4!, aser5!, aser6!)
DECLARE SUB gigi (fin1, FIN2!, fin3!, fin4!, fin5!, fin6!,
fin7!, fin8!, fin9!, fin10!, ser1!, ser2!, ser3!, ser4!,
ser5!, ser6!, afin1!(), afin2!(), afin3!(), afin4!(),
afin5!(), afin6!(), afin7!(), afin8!(), afin9!(), afin10!(),
aser1!(), aser2!(), aser3!(), aser4!(), aser5!(), aser6!(),
t!, jf1!, jf3!, jf5!, jf7!, jf9!, js1!, js3!, js5!, gig!)
DECLARE SUB AL (c$, page, oh, op, i, nof, gig!, gxxee!(),
gxx1e!(), fin1!, fin3!, fin5!, fin7!, fin9!, ser1!, ser3!,
ser5!, axee!(), axx!(), afin1!(), afin2!(), afin3!(),
afin4!(), afin5!(), afin6!(), afin7!(), afin8!(), afin9!(),
afin10!(), aser1!(), aser2!(), aser3!(), aser4!(), aser5!(),
aser6!(), uyu!, tete!, t!, gig1!, ece!, dede!, yeni!,
gxxeei!(), gxx1ei!(), axeei!(), axxi!(), tete1!, murat1!,
uyu1!, nof1!, murat!, kesin!, yesin!)
DECLARE SUB p1p11 (gxxee(), gxxeei!(), nof!, nof1!, gxxe!(),
gxx1!(), gyye!(), gyy1!(), gzze!(), gzz1!(), hj!)
DECLARE SUB p1p12 (gxxee(), gxxeei!(), page!, nof!, nof1!,
qxxe!(), qxx!(), qyye!(), qyy!(), qzze!(), qzz!(), hj1!)
DECLARE SUB p2p11 (gxxee(), gxxeei!(), page!, nof1!, nof!,
qxxe!(), qxx!(), qyye!(), qyy!(), qzze!(), qzz!(), hj2!)
DECLARE SUB p1p2 (gxxe(), gxx1!(), gyye!(), gyy1!(), gzze!(),
gzz1!(), hj1!)
DECLARE SUB p1p2g (gxxe(), gxx1!(), gyye!(), gyy1!(),
gzze!(), gzz1!(), i!, a!)
DECLARE SUB p2p12 (qxxe(), qxx!(), qyye!(), qyy!(), qzze!(),
qzz!(), i!, a!)
DECLARE SUB col (gxxee(), gxx1e!(), fin1!, fin3!, fin5!,
fin7!, fin9!, ser1!, ser3!, ser5!)
DECLARE SUB col1 (gxxeei(), c$, page!, fin1!, fin3!, fin5!,
fin7!, fin9!, ser1!, ser3!, ser5!, gxxee!())
```

```
pope = 1
op = 0
oh = 0
cc = 0

OPEN "graph1.dat" FOR INPUT AS #1
INPUT #1, g1w1, g1w2, g1w3, g1w4, g1w5, g1w6
OPEN "nooo.dat" FOR INPUT AS #2
INPUT #2, nofloor
OPEN "lagmaind.dat" FOR INPUT AS #3
INPUT #3, g1f12, g1f23, g1f34, g1f45, g2f12, g2f23, lag1,
lag2, lag3, g3111, g3112, g3112, g3111, g3121, g3122, g3131,
g3132, lag4, lag5, lagi, nof, nof1, lagn, lagra, lain, lawa,
laf1, lace, laex, lali, lame, lael
OPEN "graph2.dat" FOR INPUT AS #4
INPUT #4, g2w1, g2w2, g2w3, dum1, dum2, g2w11
OPEN "graph3.dat" FOR INPUT AS #5
INPUT #5, g3f1, g3r1, g3c1, g3f2, g3r2, g3c2, g3f3, g3r3,
g3c3
OPEN "graph4.dat" FOR INPUT AS #6
INPUT #6, g4w1, g4w2
OPEN "graphi4.dat" FOR INPUT AS #8
INPUT #8, g4wi1, g4wi2
OPEN "graphra.dat" FOR INPUT AS #9
INPUT #9, gra
OPEN "du.dat" FOR INPUT AS #10
INPUT #10, du1, du2, y1, y2
OPEN "graphfi.dat" FOR INPUT AS #11
INPUT #11, ifi, wfi, ffi, cfi, efi
OPEN "graphs.dat" FOR INPUT AS #12
INPUT #12, lise, mese, elese
OPEN "bar.dat" FOR INPUT AS #13
INPUT #13, bar

YY = nofloor / 2
ww = INT(YY)
gig = nof + 2
gig1 = nof1 + 2

IF y1 = 1 AND y2 = 2 THEN
yi = du1
ye = du2
ELSEIF y2 = 1 AND y1 = 2 THEN
yi = du2
ye = du1
ELSEIF y2 = 1 AND y1 = 1 THEN
yi = du2 + du1
ELSEIF y2 = 2 AND y1 = 2 THEN
ye = du2 + du1
END IF

g1a = (g1w1 + g1w2) * 40
g1b = (g1w3) * 40
g1c = (g1w4 + g1w5 + g1w6) * 40

g2a = g2w1 * 40
g2b = g2w2 * 40
```

```
g2c = g2w3 * 40
g2a1 = g2w11 * 40
gras = gra * 40
gras1 = gras

lag11 = lag1 * 8
lag22 = lag2 * 8
lag33 = lag3 * 8

lag22 = 0
' lag44 = lag4 * 8 no use so I put it like that '

lag55 = lag5 * 8
lag5i = lagi * 8
lagnn = lagn * 8
lagras = lagra * 8
lain = lain * 8
lawa = lawa * 8
lafl = lafl * 8
lace = lace * 8
laex = laex * 8
lali = lali * 8
lame = lame * 8
lael = lael * 8

'********SITE SET UP**************

g1yy = g1f12 * 8
g1zz = (g1f12 + g1f23) * 8
g1tt = (g1f12 + g1f23 + g1f34) * 8
g1uu = (g1f12 + g1f23 + g1f34 + g1f45) * 8

g1(1) = g1yy
g1(2) = g1zz
g1(3) = g1tt
g1(4) = g1uu

g1mini = 0
FOR i = 1 TO 4
g1mi = g1mini - g1(i)
IF g1mi <= 0 THEN
g1min = g1mini
ELSE
g1min = g1(i)
END IF
NEXT
g1xxe = -g1min
g1xx1 = g1xxe + g1a
g1yye = g1yy - g1min
g1yy1 = g1yye + g1b
g1zze = g1zz - g1min
g1zz1 = g1zze + g1c
g1tte = g1tt - g1min
g1tt1 = g1tte + g1d
g1uue = g1uu - g1min
g1uu1 = g1uue + g1w9 * 40
```

```
'************FOUNDATIONS*****************
g22b = g1zze + lagnn
g22b1 = g22b + g2a1

g2xxe = g22b + lag11
g2xx1 = g2xxe + g2a

g2yye = g2xxe + g2f12 * 8
IF g2yye > g2xx1 THEN
g2yye = g2xx1
END IF
g2yy1 = g2yye + g2b

g2zze = g2yye + 8 * g2f23
IF g2zze > g2yy1 THEN
g2zze = g2yy1
END IF
g2zz1 = g2zze + g2c


'**********************GROUND FLOOR*******************
DIM gxxe(50), gxx1(50), gyye(50), gyy1(50), gzze(50),
gzz1(50), gxxee(50), gxx1e(50)
DIM axee(50), axx(50), qxxe(50), qxx(50), qyye(50), qyy(50),
qzze(50), qzz(50), axeei(50), axxi(50), gxxeei(50),
gxx1ei(50)
DIM qxx1(50), qyy1(50), qzz1(50), afin1(50), afin2(50),
afin3(50), afin4(50), afin5(50), afin6(50), afin7(50),
afin8(50), afin9(50), afin10(50), aser1(50), aser2(50),
aser3(50), aser4(50), aser5(50), aser6(50), z%(50)

'ground slab'
g2tte = 0
gyye(44) = g2zze + lag22 + lag33
IF gyye(44) > g2zz1 THEN
gyye(44) = g2zz1
END IF

gyy1(44) = gyye(44) + 40 * (g3c2) + 40 / 5 * (g3l21 + g3l22)
gxxe(4) = gyye(44) + g3l11 * 40 / 5
gxx1(4) = gxxe(4) + 40 * (g3c1) + 40 / 5 * (g3l11 + g3l12)
gyye(4) = gxxe(4) + g3l11 * 40 / 5

gyy1(4) = gyye(4) + 40 * (g3c2) + 40 / 5 * (g3l21 + g3l22)
gzz1(4) = gyye(4) + g3l12 * 40 / 5
gzze(4) = gzz1(4) + 40 * (g3c3) + 40 / 5 * (g3l31 + g3l32)

IF nof = 1 THEN
gxxee(4) = gxxe(4) + lag55
IF ye > (g4w1 + g4w2) THEN
gxx1e(4) = gxxee(4) + 40 * ye
ELSE
gxx1e(4) = gxxee(4) + 40 * (g4w1 + g4w2) * nofloor
END IF
```

```
fin6 = fin5 + 40 * ffi
fin7 = gxxee(4) + lace
fin8 = fin7 + 40 * cfi
fin9 = gxxee(4) + laex
fin10 = fin9 + 40 * efi
ser1 = gxxee(4) + lali
ser2 = ser1 + 40 * lise
ser3 = gxxee(4) + lame
ser4 = ser3 + 40 * mese
ser5 = gxxee(4) + lael
ser6 = ser5 + 40 * elese
END IF
IF i = nof1 + 3 THEN
gxxeei(4) = gzze(i) + lag5i
IF yi > (g4wi1 + g4wi2) THEN
gxx1ei(4) = gxxeei(4) + 40 * yi
ELSE
gxx1ei(4) = gxxeei(4) + 40 * (g4wi1 + g4wi2) * nofloor
END IF
END IF
NEXT


'***************SCROLLING LEFT/RIGHT/UP/DOWN****************'
MINKEY(3) = 0: MAXKEY(3) = 0: EXTMINKEY(3) = 0:
 EXTMAXKEY(3) = 0 EXTSPECIALKEY$(3) = CHR$(69) + CHR$(72) +
CHR$(75) + CHR$(80) + CHR$(77)
TRUE = -1: FALSE = 0: CAPSON(3) = TRUE: CALLER = 3:
SPECIALKEY$(3) = CHR$(27)

SCREEN 12
WIDTH 80, 60
page = 1
a = 1
PAGE1 = 0
t = 0
tt = 0
'new'
tuf = 0

110 VIEW (0, 0)-(621, 479): CLS 1

LOCATE 2, 1
PRINT , TAB(80); "weeks"
COLOR 12
LOCATE 5, 1
PRINT , TAB(80); "SUBSTRUCTURE";
COLOR 15
LOCATE 9, 1
COLOR 3
PRINT "1";
COLOR 15
PRINT " Excavate"; TAB(480); "  Topsoil ";
LOCATE 13, 1
COLOR 3
PRINT "2";
COLOR 15
PRINT " Reduce       "; TAB(480); "  Level  ";
```

```
LOCATE 17, 1
COLOR 3
PRINT "3";
COLOR 15
PRINT " Excavate / "; TAB(480); "  Compact "; TAB(480); "
Fdns "; TAB(480); "       "
LOCATE 23, 1
COLOR 3
PRINT "4";
COLOR 15
PRINT " Blinding  "; TAB(480); "  Fdns       "
LOCATE 27, 1
COLOR 3
PRINT "5";
COLOR 15
PRINT " Formwork  "; TAB(480); "  Fdns       "
LOCATE 31, 1
COLOR 3
PRINT "6";
COLOR 15
PRINT " Reinforcem "; TAB(480); "  Fdns       "
LOCATE 35, 1
COLOR 3
PRINT "7";
COLOR 15
PRINT " Conc Fdn"
LOCATE 39, 1
COLOR 12
PRINT , TAB(480); "GROUND SLAB"
COLOR 3
LOCATE 43, 1
PRINT "8";
COLOR 15
PRINT " Gr Flr"; TAB(480); "  Frc Slab/ "; TAB(480); "
Beams"
COLOR 12
LOCATE 49, 1
PRINT "SUPERSTRUCT."
COLOR 13
LOCATE 51, 1
PRINT , TAB(480); " GROUND TO", TAB(480); " 1st FLOOR ";
COLOR 3
LOCATE 55, 1
PRINT "9";
COLOR 15
PRINT " Frc Cols/", TAB(480); "  Walls";

LINE (100, 8)-(100, 462)
LINE (0, 30)-(620, 30)
LINE (0, 8)-(621, 9), , BF
LINE (140, 10)-(141, 35), , BF
LINE (180, 10)-(181, 35), , BF
LINE (220, 10)-(221, 35), , BF
LINE (260, 10)-(261, 35), , BF
LINE (300, 10)-(301, 35), , BF
LINE (340, 10)-(341, 35), , BF
LINE (380, 10)-(381, 35), , BF
LINE (420, 10)-(421, 35), , BF
```

```
LINE (460, 10)-(461, 35), , BF
LINE (500, 10)-(501, 35), , BF
LINE (540, 10)-(541, 35), , BF
LINE (580, 10)-(581, 35), , BF
LINE (620, 10)-(621, 35), , BF


10 IF page < 0 OR PAGE1 < 0 OR t < 0 THEN
GOTO 10000
END IF
VIEW (100, 30)-(621, 479): CLS 1
LINE (0, 0)-(522, 0)
LINE (0, 432)-(522, 432)
IF bar = 1 THEN
FOR i = 0 TO 520 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF
IF page = 1 AND PAGE1 = 0 THEN

11 CALL p1p10(g1xxe, g1xx1, g1yye, g1yy1, g1zze, g1zz1,
g1tte, g1tt1, g1uue, g1uu1, g22b, g22b1, g2xxe, g2xx1, g2yye,
g2yy1, g2zze, g2zz1, gyye(), gyy1(), gxxe(), gxx1())
END IF

111 IF page = 1 THEN
FOR i = 1 TO 9
LOCATE 3, 15 + (i - 1) * 5: PRINT i;
NEXT
LOCATE 3, 60: PRINT "10"
LOCATE 3, 65: PRINT "11"
LOCATE 3, 70: PRINT "12"
LOCATE 3, 75: PRINT " 13"
ELSE
GOTO 123
END IF
CALL keyin(MINKEY(), MAXKEY(), EXTMINKEY(), EXTMAXKEY(),
SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON(), CALLER, XFLD,
YFLD, LNGTH, TRUE, FALSE, c$)

'*****************print screen*******************************

IF ASC(c$) = 27 THEN
CLOSE
GOTO 10000
END IF
IF ASC(c$) = 75 THEN
GOTO 10000
ELSEIF ASC(c$) = 80 THEN
PAGE1 = PAGE1 + 1
GOTO 889
ELSEIF ASC(c$) = 72 THEN
PAGE1 = PAGE1 - 1
IF PAGE1 < 0 OR a < 0 OR t < 0 OR page = 0 THEN
GOTO 10000
ELSEIF PAGE1 = 0 THEN
GOTO 110
ELSEIF PAGE1 = 1 THEN
```

```
GOTO 888
ELSEIF PAGE1 = 2 THEN
GOTO 81
ELSEIF PAGE1 > 2 THEN
GOTO 9999
END IF
ELSEIF ASC(c$) = 77 THEN
t = 0
tuf = 0
GOTO 777
ELSE
GOTO 10000
END IF

777 t = t + 1
page = page + 1
tuf = tuf + 1

77 IF t < 0 OR page = 0 OR PAGE1 < 0 OR a > nofloor + 1 THEN
GOTO 10000
END IF

IF gyy1(44) >= 480 * t THEN
qyye(44) = gyye(44) - 480 * t
qyy1(44) = gyy1(44) - 480 * t
ELSEIF gyy1(44) < 480 * t THEN
qyye(44) = 0
qyy1(44) = 0
END IF

gig = gig + 1

IF gxxee(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t

jason = t + 1

DO WHILE axee(gig) >= 480 * jason
axee(gig) = axee(gig) - 480 * jason
jason = jason + 1
LOOP
axx(gig) = gxx1e(4) - 480 * t

ELSEIF gxxee(4) < 480 * t AND gxx1e(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t
axx(gig) = gxx1e(4) - 480 * t

ELSEIF gxx1e(4) < 480 * t THEN
axee(gig) = 0
axx(gig) = 0
END IF

CALL gigi(fin1, FIN2, fin3, fin4, fin5, fin6, fin7, fin8,
fin9, fin10, ser1, ser2, ser3, ser4, ser5, ser6, afin1(),
afin2(), afin3(), afin4(), afin5(), afin6(), afin7(),
afin8(), afin9(), afin10(), aser1(), aser2(), aser3(),
aser4(), aser5(), aser6(), t, jf1, jf3, jf5, jf7, jf9, js1,
js3, js5, gig)
```

```
gig1 = gig1 + 1

IF gxxeei(4) >= 480 * t THEN
axeei(gig1) = gxxeei(4) - 480 * t

jason1 = t + 1


DO WHILE axeei(gig1) >= 480 * jason1
axeei(gig1) = axeei(gig1) - 480 * jason1
jason1 = jason1 + 1
LOOP
axxi(gig1) = gxx1ei(4) - 480 * t
ELSEIF gxxeei(4) < 480 * t AND gxx1ei(4) >= 480 * t THEN
axeei(gig1) = gxxeei(4) - 480 * t
axxi(gig1) = gxx1ei(4) - 480 * t
ELSEIF gxx1ei(4) < 480 * t THEN
axeei(gig1) = 0
axxi(gig1) = 0
END IF

VIEW (100, 30)-(621, 479): CLS 1
LINE (0, 432)-(521, 432)
LINE (0, 0)-(521, 0)
LINE (0, 0)-(2, 432), , BF
IF bar = 1 THEN
FOR i = 1 TO 521 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF

123 FOR k = 1 TO 12
LOCATE 3, 9 + 5 * k: PRINT (t) * 12 + k
NEXT
LOCATE 3, 9 + 5 * k + 1: PRINT (t) * 12 + k
IF g1xx1 >= 480 * t THEN
xxe = g1xxe - 480 * t
xx = g1xx1 - 480 * t
ELSEIF g1xx1 < 480 * t THEN
xxe = 0
xx = 0
END IF
IF g1yy1 >= 480 * t THEN
yye = g1yye - 480 * t
YY = g1yy1 - 480 * t
ELSEIF g1yy1 < 480 * t THEN
yye = 0
YY = 0
END IF
IF g1zz1 >= 480 * t THEN
zze = g1zze - 480 * t
zz = g1zz1 - 480 * t
ELSEIF g1zz1 < 480 * t THEN
zze = 0
zz = 0
END IF
IF g1tt1 >= 480 * t THEN
```

```
tte = g1tte - 480 * t
tt = g1tt1 - 480 * t
ELSEIF g1tt1 < 480 * t THEN
tte = 0
tt = 0
END IF
IF g1uu1 >= 480 * t THEN
uue = g1uue - 480 * t
uu = g1uu1 - 480 * t
ELSEIF g1uu1 < 480 * t THEN
uue = 0
uu = 0
END IF

IF g2xx1 >= 480 * t THEN
xxe2 = g2xxe - 480 * t
xx2 = g2xx1 - 480 * t
ELSEIF g2xx1 < 480 * t THEN
xxe2 = 0
xx2 = 0
END IF

IF g22b1 >= 480 * t THEN
xxxe2 = g22b - 480 * t
xxx2 = g22b1 - 480 * t
ELSEIF g22b1 < 480 * t THEN
xxxe2 = 0
xxx2 = 0
END IF

IF g2yy1 >= 480 * t THEN
yye2 = g2yye - 480 * t
yy2 = g2yy1 - 480 * t
ELSEIF g2yy1 < 480 * t THEN
yye2 = 0
yy2 = 0
END IF

IF g2zz1 >= 480 * t THEN
zze2 = g2zze - 480 * t
zz2 = g2zz1 - 480 * t
ELSEIF g2zz1 < 480 * t THEN
zze2 = 0
zz2 = 0
END IF

IF g2tt1 >= 480 * t THEN
tte2 = g2tte - 480 * t
tt2 = g2tt1 - 480 * t
ELSEIF g2tt1 < 480 * t THEN
tte2 = 0
tt2 = 0
END IF

IF gxx1(4) >= 480 * t THEN
qxxe(4) = gxxe(4) - 480 * t
qxx(4) = gxx1(4) - 480 * t
ELSEIF gxx1(4) < 480 * t THEN
```

```
qxxe(4) = 0
qxx(4) = 0
END IF

IF gyy1(4) >= 480 * t THEN
qyye(4) = gyye(4) - 480 * t
qyy(4) = gyy1(4) - 480 * t
ELSEIF gyy1(4) < 480 * t THEN
qyye(4) = 0
qyy(4) = 0
END IF

IF gzze(4) >= 480 * t THEN
qzze(4) = gzz1(4) - 480 * t
qzz(4) = gzze(4) - 480 * t
ELSEIF gzze(4) < 480 * t THEN
qzze(4) = 0
qzz(4) = 0
END IF

FOR gig = nof + 3 TO nofloor + 4
IF gxxee(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t

jason = t + 1

DO WHILE axee(gig) >= 480 * jason
axee(gig) = axee(gig) - 480 * jason
jason = jason + 1
LOOP

axx(gig) = gxx1e(4) - 480 * t
ELSEIF gxxee(4) < 480 * t AND gxx1e(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t
axx(gig) = gxx1e(4) - 480 * t
ELSEIF gxx1e(4) < 480 * t THEN
axee(gig) = 0
axx(gig) = 0
END IF
NEXT

CALL ngigi(fin1, FIN2, fin3, fin4, fin5, fin6, fin7, fin8,
fin9, fin10, ser1, ser2, ser3, ser4, ser5, ser6, afin1(),
afin2(), afin3(), afin4(), afin5(), afin6(), afin7(),
afin8(), afin9(), afin10(), aser1(), aser2(), aser3(),
aser4(), aser5(), aser6(), t, jf1, jf3, jf5, jf7, jf9, js1,
js3, js5, gig, nofloor)

FOR gig1 = nof1 + 3 TO nofloor + 4
IF gxxeei(4) >= 480 * t THEN
axeei(gig1) = gxxeei(4) - 480 * t

jason1 = t + 1

DO WHILE axeei(gig1) >= 480 * jason1
axeei(gig1) = axeei(gig1) - 480 * jason1
jason1 = jason1 + 1
LOOP
```

```
axxi(gig1) = gxx1ei(4) - 480 * t
ELSEIF gxxeei(4) < 480 * t AND gxx1ei(4) >= 480 * t THEN
axeei(gig1) = gxxeei(4) - 480 * t
axxi(gig1) = gxx1ei(4) - 480 * t
ELSEIF gxx1ei(4) < 480 * t THEN
axeei(gig1) = 0
axxi(gig1) = 0
END IF
NEXT


FOR i = 5 TO nofloor + 4
IF gxx1(i) >= 480 * t THEN
qxxe(i) = gxxe(i) - 480 * t
qxx(i) = gxx1(i) - 480 * t
ELSEIF gxx1(i) < 480 * t THEN
qxxe(i) = 0
qxx(i) = 0
END IF
IF gyy1(i) >= 480 * t THEN
qyye(i) = gyye(i) - 480 * t
qyy(i) = gyy1(i) - 480 * t
ELSEIF gyy1(i) < 480 * t THEN
qyye(i) = 0
qyy(i) = 0
END IF

IF gzze(i) >= 480 * t THEN
qzze(i) = gzz1(i) - 480 * t
qzz(i) = gzze(i) - 480 * t
ELSEIF gzze(i) < 480 * t THEN
qzze(i) = 0
qzz(i) = 0
END IF
NEXT

VIEW (100, 30)-(621, 479): CLS 1
LINE (0, 432)-(521, 432)
LINE (0, 0)-(521, 0)
LINE (0, 0)-(2, 432), , BF
IF bar = 1 THEN
FOR i = 1 TO 521 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF

IF page >= 2 AND PAGE1 = 0 AND a < nofloor + 1 THEN
CALL p1p10e(xxe, xx, yye, YY, zze, zz, tte, tt, uue, uu,
xxxe2, xxx2, xxe2, xx2, yye2, yy2, zze2, zz2, qyye(), qyy1(),
qxxe(), qxx())

ELSEIF page >= 2 AND PAGE1 = 1 AND a < nofloor + 1 THEN
CALL p2p11(gxxee(), gxxeei(), page, nof1, nof, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), hj2)
ELSEIF page = 1 AND PAGE1 = 1 THEN
CALL p1p11(gxxee(), gxxeei(), nof, nof1, gxxe(), gxx1(),
gyye(), gyy1(), gzze(), gzz1(), hj)
```

```
ELSEIF page = 1 AND PAGE1 = 2 THEN
CALL p1p12(gxxee(), gxxeei(), page, nof, nof1, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), hj1)

ELSEIF page = 1 AND PAGE1 > 2 AND a < nofloor + 1 THEN
CALL p1p12g(gxxe(), gxx1(), gyye(), gyy1(), gzze(), gzz1(),
i, a)

ELSEIF page >= 2 AND PAGE1 >= 2 AND a < nofloor + 1 AND exc =
1 AND yeni = 1 THEN
i = a + 3
COLOR 5
LINE (qxxe(i), 80)-(qxx(i), 85), , BF
LINE (qyye(i), 110)-(qyy(i), 115), , BF
COLOR 4
LINE (qyye(i), 80)-(qyye(i) + 1, 110), , BF
COLOR 15
IF gra1 < 0 THEN
GOTO 445
END IF
IF qyye(i) = 0 THEN
grae = gyye(i) + lagras
ELSE
grae = qyye(i) + lagras
END IF
gra1 = grae + gras

IF qyye(i - 1) = 0 THEN
cvont = cvont + 1
END IF

IF gras1 < 0 AND ASC(c$) = 77 THEN
dd = dd + 1
END IF
IF ASC(c$) = 75 THEN
dd = dd - 1
END IF

445 COLOR 5
IF RA1 < 0 AND grae = 0 AND gras1 < 0 AND ASC(c$) = 77 THEN
LINE (0, 260)-(0, 265), , BF
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
ELSEIF gra1 < 480 * tuf AND gras1 < 0 AND dd = 1 AND grae < 0
THEN
LINE (0, 260)-(grae, 265), , BF
```

```
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)

ELSEIF gra1 < 480 * tuf AND gras1 <= gras AND grae > 0 THEN

IF gra1 > 0 AND gras1 <> gras AND gras1 >= 0 AND
cvont = 0 THEN
        LINE (grae - 200, 260)-(gra1, 265), , BF
        COLOR 10
        LINE (grae - 200, 110)-(grae - 200 + 1, 260), , BF

CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)

        ELSEIF gra1 > 0 AND gras1 <> gras AND gras1 >= 0 AND
cvont <> 0 THEN
        LINE (grae - 480 * (tuf - 1), 260)-(gra1 - 480 * (tuf
- 1), 265), , BF
        COLOR 10
        LINE (grae - 480 * (tuf - 1), 110)-(grae + 1 - 480 *
(tuf - 1), 260), , BF

CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
```

```
        ELSEIF gra1 > 0 AND gras1 <> gras AND gras1 < 0 THEN
                IF dd <= 0 THEN 'dd< new'
        LINE (0, 260)-(gra1 - 480, 265), , BF
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)

                ELSEIF dd = 1 THEN

        GRA11 = gra1
                IF GRA11 < 480 THEN
        GOTO 5555
                END IF
        DO UNTIL GRA11 < 0
        GRA11 = GRA11 - 480
        LOOP
GRA11 = GRA11 + 480

5555 COLOR 5
IF GRA11 < 480 * (tuf) THEN
IF gra1 < 480 * tuf AND gra1 > 0 THEN
GRA15 = 0
ELSEIF gra1 < 0 THEN
GRA15 = gra1
END IF
END IF

IF GRA15 < 0 AND GRA11 > 0 AND gra1 < 480 * tuf THEN
LINE (0, 260)-(0, 265), , BF
ELSE
LINE (0, 260)-(GRA11, 265), , BF
END IF

1010 COLOR 5
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
```

```
                ELSEIF dd > 1 THEN

        LINE (0, 260)-(0, 265), , BF
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tetel,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)

                END IF

        ELSEIF gra1 > 0 AND gras1 = gras AND dd > 0 THEN

        LINE (grae, 260)-(gra1, 265), , BF
        COLOR 10
        LINE (grae, 110)-(grae + 1, 260), , BF

        ELSEIF gra1 < 0 AND gras1 < gras THEN

                IF gra1 + 480 < 0 THEN
        LINE (0, 260)-(0, 265), , BF

                ELSE
        LINE (grae - 480, 260)-(gra1 + 480, 265), , BF
        COLOR 10
        LINE (grae - 480, 110)-(grae - 480 + 1, 260), , BF

                END IF

        ELSEIF gra1 > 480 AND gras = gras1 THEN
        IF dd > 1 THEN
        LINE (0, 260)-(GRA11, 265)

        ELSEIF dd <= 0 THEN
        LINE (gra1, 260)-(grae, 265), , BF
        COLOR 10
        LINE (gra1, 110)-(gra1 + 1, 260), , BF

        END IF

        END IF
ELSE
COLOR 5
IF gras1 >= gras AND qyye(i - 1) > 0 THEN

LINE (grae, 260)-(gra1, 265), , BF

        COLOR 10
        LINE (grae, 110)-(grae + 1, 260), , BF
IF gxxee(4) < 480 * (page - 1) THEN
```

```
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
END IF
ELSEIF gras1 >= gras AND qyye(i - 1) = 0 THEN
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
COLOR 5
LINE (0, 260)-(gra1 - 480, 265), , BF
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)

ELSEIF gras1 < gras AND gra1 < 0 AND grae <= 0 AND dd > 1
THEN
grae1 = grae
GRA11 = gra1
DO UNTIL grae1 >= 480
grae1 = grae1 + 480
LOOP
DO UNTIL GRA11 >= 480
GRA11 = GRA11 + 480
LOOP

LINE (grae1, 260)-(GRA11, 260)
COLOR 10
        LINE (grae1, 110)-(grae1 + 1, 260), , BF
```

```
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)


ELSEIF cvot = 0 THEN
LINE (0, 260)-(gra1 - 480, 265), , BF
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)


ELSEIF cvot <> 0 THEN
LINE (0, 260)-(grae, 265), , BF
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
END IF
END IF
IF gra1 < 0 THEN
GOTO 4451
END IF

grae = qyye(i) + lagras
gra1 = grae + gras

4451 COLOR 5
IF gra1 < 0 AND grae >= 0 AND ASC(c$) = 77 THEN
```

```
LINE (0, 260)-(0, 265), , BF
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, aser3, aser4,
aser5, aser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)


ELSEIF gra1 < 0 AND grae < 0 THEN
LINE (grae, 260)-(gra1 + 480, 265), , BF
        COLOR 10
        LINE (grae, 110)-(grae + 1, 260), , BF

CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, aser3, aser4,
aser5, aser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)

ELSEIF grae < 0 AND gra1 > 480 THEN
LINE (0, 260)-(gra1, 265), , BF
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, aser3, aser4,
aser5, aser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)

ELSEIF grae > 0 AND gra1 > 0 AND vv <> 1 THEN

CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
```

```
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
vv = 1
END IF
COLOR 15

8989 ELSEIF page >= 2 AND PAGE1 >= 2 AND a < nofloor + 1 THEN
IF dede = 1 THEN
CALL p2p22(gxxee(), gxxeei(), nof, nof1, page, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), i, a)
ELSE
CALL p2p22(gxxee(), gxxeei(), nof, nof1, page, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), i, a)

IF a = nofloor AND nofloor / 2 <> ww THEN
LINE (0, 410)-(0, 415), , BF
ELSE
COLOR 5
LINE (qzze(i + 2), 410)-(qzz(i + 2), 415), , BF
COLOR 15
END IF
END IF
END IF

222 CALL keyin(MINKEY(), MAXKEY(), EXTMINKEY(), EXTMAXKEY(),
SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON(), CALLER, XFLD,
YFLD, LNGTH, TRUE, FALSE, c$)
IF ASC(c$) = 27 THEN
CLOSE
GOTO 10000
END IF
IF ASC(c$) = 75 THEN

t = t - 1
page = page - 1
oh = oh + 1
kesin = kesin + 1
IF PAGE1 >= nofloor - 4 THEN
tuf = tuf - 1
gra1 = gra1 + 480
gras1 = gras1 + 480
IF gra1 < 480 * tuf THEN
gra1 = gra1 + 480
END IF
END IF

IF yeni <> 1 AND yeni1 <> 1 THEN
yeni = 0
END IF
        IF page = 0 OR t < 0 OR PAGE1 < 0 THEN
GOTO 10000
        ELSEIF page = 1 THEN
FOR i = 1 TO 9
LOCATE 3, 15 + (i - 1) * 5: PRINT i;
NEXT
```

```
LOCATE 3, 60: PRINT "10 "
LOCATE 3, 65: PRINT "11 "
LOCATE 3, 70: PRINT "12 "
LOCATE 3, 75: PRINT "13   "
                IF page = 1 AND PAGE1 = 0 THEN
GOTO 10
                ELSEIF page = 1 AND PAGE1 = 1 THEN
GOTO 889
                ELSEIF page = 0 OR PAGE1 < 0 OR t < 0 THEN
GOTO 10000
                ELSE
 GOTO 123
                END IF
GOTO 889
        ELSE
GOTO 77
        END IF

ELSEIF ASC(c$) = 77 THEN
op = op + 1
yeni1 = 1
yesin = yesin + 1
IF PAGE1 >= nofloor - 4 THEN
gra1 = gra1 - 480
gras1 = gras1 - 480
END IF

IF gra1 < 480 * tuf AND gra1 > 0 THEN
gra1 = gra1 - 480
ELSEIF gra1 < 0 THEN
GRA15 = gra1
END IF
GOTO 777
ELSEIF ASC(c$) = 80 THEN
PAGE1 = PAGE1 + 1
IF PAGE1 = 1 THEN
GOTO 889
ELSEIF PAGE1 = 2 THEN
GOTO 81
ELSE
cc = cc - 3
GOTO 999
END IF

ELSEIF ASC(c$) = 72 THEN
yeni = 0
PAGE1 = PAGE1 - 1
IF PAGE1 < 0 OR a < 0 OR t < 0 THEN
GOTO 10000
ELSEIF PAGE1 = 1 AND a > 0 THEN
GOTO 889
ELSEIF PAGE1 = 2 AND a > 0 THEN
GOTO 81
ELSEIF PAGE1 = 0 AND a > 0 THEN
GOTO 110
ELSEIF a = 1 THEN
GOTO 889
ELSEIF a = nofloor AND nofloor / 2 <> ww AND dff = 0 THEN
```

```
a = a + 4
cc = cc + 4
GOTO 9999
ELSEIF a = nofloor AND nofloor / 2 <> ww AND dff = 1 THEN
a = a
cc = cc + 20
GOTO 9999
ELSE
a = a + 1
cc = cc + 1
IF a = 0 THEN
GOTO 889
ELSEIF a < 0 THEN
GOTO 10000
ELSE
GOTO 9999
END IF
END IF
ELSE
GOTO 10000
END IF

888 IF PAGE1 = 1 THEN
889 a = 1
IF t < 0 THEN
GOTO 10000
END IF
VIEW (100, 30)-(621, 479): CLS 1
COLOR 12
LOCATE 5, 2
PRINT , TAB(480); "SUPERSTRUCT. "
COLOR 13
LOCATE 8, 1
PRINT , TAB(480); "                "
LOCATE 9, 1
PRINT , TAB(480); " GROUND TO", TAB(480); " 1st FLOOR ";
COLOR 15
LOCATE 12, 1
PRINT , TAB(480); "                "
LOCATE 13, 1
COLOR 3
PRINT , TAB(480); " 9";
COLOR 15
PRINT " Frc Cols/ ", TAB(480); "   Walls "; TAB(480); "    "
LOCATE 17, 1
COLOR 3
PRINT , TAB(480); "10";
COLOR 15
PRINT " 1st Flr "; TAB(480); "   Frc Slab/", TAB(480);
" Beams "; TAB(480); "                                "
LOCATE 22, 1
COLOR 3
PRINT , TAB(480); "11";
COLOR 15
PRINT " Frc Stair"; TAB(480); "            "
COLOR 13
LOCATE 25, 1
```

```
PRINT , TAB(480); " 1st TO", TAB(480); " 2nd FLOOR  ";
TAB(480); "           "
COLOR 15
LOCATE 29, 1
COLOR 3
PRINT , TAB(480); "12";
COLOR 15
PRINT " Frc Cols/", TAB(480); "   Walls       "; TAB(480);" "
LOCATE 33, 1
COLOR 3
PRINT , TAB(480); "13";
COLOR 15
PRINT " 2nd Flr   "; TAB(480); "   Frc Slab/", TAB(480);
"   Beams         "
COLOR 3
LOCATE 38, 1
PRINT , TAB(480); "14";
COLOR 15
PRINT " Frc Stair"; TAB(480); "                 "
COLOR 13
LOCATE 41, 1
PRINT , TAB(480); " 2nd TO ", TAB(480); " 3rd FLOOR  ";
TAB(480); "           "
COLOR 3
LOCATE 45, 1
PRINT , TAB(480); "15";
COLOR 15
PRINT " Frc Cols/", TAB(480); "   Walls       "; TAB(480); ""
LOCATE 49, 1
COLOR 3
PRINT , TAB(480); "16";
COLOR 15
PRINT " 3rd Flr "; TAB(480); "   Frc Slab/", TAB(480);
"   Beams        "; TAB(480); "               "
COLOR 3
LOCATE 54, 1
PRINT , TAB(480); "17";
COLOR 15
PRINT " Frc Stair "; TAB(480); "         "

LINE (0, 0)-(522, 0)
LINE (0, 432)-(522, 432)
IF bar = 1 THEN
FOR i = 0 TO 520 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF
8899 IF PAGE1 = 1 AND page = 1 THEN
CALL plp11(gxxee(), gxxeei(), nof, nof1, gxxe(), gxx1(),
gyye(), gyy1(), gzze(), gzz1(), hj)

ELSEIF page >= 2 AND PAGE1 = 1 AND a < nofloor + 1 THEN
CALL p2p11(gxxee(), gxxeei(), page, nof1, nof, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), hj2)
END IF

ELSEIF page = 1 AND PAGE1 = 2 THEN
```

```
CALL plpl2(gxxee(), gxxeei(), page, nof, nof1, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), hj1)

END IF

CALL keyin(MINKEY(), MAXKEY(), EXTMINKEY(), EXTMAXKEY(),
SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON(), CALLER, XFLD,
YFLD, LNGTH, TRUE, FALSE, c$)

IF ASC(c$) = 27 THEN
CLOSE
GOTO 10000
END IF
IF page = 1 AND ASC(c$) = 75 AND t <> 0 THEN
GOTO 10
ELSEIF ASC(c$) = 75 THEN
t = t - 1
page = page - 1
'new'
IF PAGE1 >= nofloor - 4 THEN
tuf = tuf - 1
END IF
IF page = 0 OR PAGE1 < 0 OR t < 0 THEN
GOTO 10000
END IF
GOTO 77
ELSEIF ASC(c$) = 77 THEN
GOTO 777
ELSEIF ASC(c$) = 80 THEN
PAGE1 = PAGE1 + 1
IF PAGE1 = 2 THEN
GOTO 81
ELSE
GOTO 999
END IF
ELSEIF ASC(c$) = 72 THEN
PAGE1 = PAGE1 - 1
IF PAGE1 < 0 OR a < 0 THEN
GOTO 10000
ELSEIF PAGE1 = 1 THEN
GOTO 889
ELSEIF PAGE1 = 2 THEN
GOTO 81
ELSEIF PAGE1 = 0 THEN
GOTO 110
ELSE
a = a - 4
cc = cc - 4
IF a = 2 THEN
GOTO 81
ELSEIF a < 0 THEN
GOTO 10000
ELSE
GOTO 9999
END IF
END IF
END IF
```

```
999 IF a = nofloor AND ASC(c$) = 80 AND nofloor / 2 = ww THEN
GOTO 10000
END IF
IF a = nofloor + 1 AND ASC(c$) = 80 AND nofloor / 2 <> ww
THEN
GOTO 10000
END IF
IF a = nofloor AND nofloor / 2 <> ww AND ASC(c$) = 80 THEN
GOTO 1212
END IF
IF a = nofloor - 1 AND nofloor / 2 = ww AND ASC(c$) = 72 THEN
qqq = 1
END IF

IF a = nofloor - 1 AND nofloor / 2 = ww AND ASC(c$) = 80 AND
df = 1 THEN
qbb = 1
END IF

IF ASC(c$) = 27 THEN
CLOSE
GOTO 10000
END IF
9999 IF PAGE1 >= 3 AND ASC(c$) = 80 AND a = nofloor THEN
a = a
ELSEIF PAGE1 >= 3 AND ASC(c$) = 80 THEN

IF a = nofloor - 1 AND nofloor / 2 = ww AND qqq = 1 THEN
cc = cc - 3
END IF

IF a = nofloor - 1 AND nofloor / 2 = ww AND qbb = 1 THEN
cc = cc - 3
END IF
a = a - 1
cc = cc - 1
ELSEIF PAGE1 >= 3 AND ASC(c$) = 72 THEN
a = a - 3
IF nofloor / 2 <> ww THEN
cc = cc
ELSE
cc = cc - 2
END IF
END IF
IF nofloor / 2 <> ww AND ASC(c$) = 72 THEN
cc = cc - 18
END IF

IF a = 0 THEN
GOTO 888
END IF

IF nofloor / 2 = ww AND a = nofloor - 2 THEN
GOTO 1222
ELSEIF nofloor / 2 <> ww AND a = nofloor - 1 THEN
GOTO 1222
END IF
```

```
VIEW (100, 30)-(621, 479): CLS 1

IF a = nofloor - 1 THEN
a = a + 1
cc = cc - 2
GOTO 156
END IF
IF a = nofloor THEN
GOTO 156
END IF
IF ASC(c$) = 80 THEN
cc = cc + 1

ELSEIF ASC(c$) = 72 AND che = 1 THEN
cc = cc - 20
ELSE
cc = cc - 8
END IF

LOCATE (5), 1
PRINT , TAB(480); a
PRINT , TAB(480); "                ";
COLOR 13
LOCATE (9), 1
PRINT , TAB(480); a; "th TO "; TAB(480); a + 1; "th FLOOR"
LOCATE (13), 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Frc Cols/", TAB(480); "    Walls       "; TAB(480); ""
LOCATE (17), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT a + 1; "th Flr",TAB(480); "    Frc Slab/",
TAB(480); "    Beams";
TAB(480); "          "
LOCATE (22), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Frc Stair";TAB(480); "                                ";
TAB(480); "                        "

IF a >= nofloor - 1 AND nofloor / 2 = ww THEN
COLOR 13
LOCATE (25), 1
PRINT , TAB(480); a; "th TO "; TAB(480); " ROOF          "
LOCATE (29), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Frc Cols/", TAB(480); "   Walls";
LOCATE (33), 1
```

```
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT " Roof                ";  TAB(480); "   Frc Slab/",
TAB(480); "   Beams         ";
LOCATE (38), 1
COLOR 12
PRINT , TAB(480); "BUILDING  ", TAB(480); "ENVELOPE "
LOCATE (42), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT , TAB(480); "Internal    ", TAB(480); "Walls    "
LOCATE (46), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT , TAB(480); "External    ", TAB(480); "Walls      "
LOCATE (50), 1
COLOR 12
PRINT , TAB(480); "ROOFING          ", TAB(480); "       "
COLOR 13
LOCATE (53), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT , TAB(480); "Roof Asphalt", TAB(480); "                "
a = a + 1
cc = cc + 1
ece = 1
GOTO 156
ELSE
a = a + 1
COLOR 13
LOCATE (25), 1
PRINT TAB(480); a; "th TO "; TAB(480); a + 1; "th FLOOR",
TAB(480); "                    "
LOCATE (29), 1
COLOR 3
cc = cc + 1
PRINT cc;
COLOR 15
PRINT "Frc Cols/", TAB(480); "    Walls   "; TAB(480); "   ";
TAB(480); "                   "
LOCATE (33), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT a + 1; "th Flr"; TAB(480); "    Frc Slab/", TAB(480);
"    Beams";
TAB(480); "              "; TAB(480); "                "
LOCATE (38), 1
cc = cc + 1
```

```
COLOR 3
PRINT cc;
COLOR 15
PRINT "Frc Stair"; TAB(480); "                ";
END IF
a = a + 1

IF a >= nofloor - 1 THEN
LOCATE (40), 1
PRINT , TAB(480); "                                "
LOCATE (41), 7
COLOR 13
PRINT , TAB(480); a; "th TO "; TAB(480); " ROOF                "
LOCATE (45), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15

PRINT "Frc Cols/", TAB(480); "    Walls         ";TAB(480); ""
LOCATE (49), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Roof        "; TAB(480); "    Frc Slab/",
TAB(480); "    Beams               "; TAB(480); "              "
FOR G = 53 TO 57
LOCATE (G), 1
PRINT , TAB(480); "                "
NEXT
a = a + 1
cc = cc + 1
dede = 1
GOTO 156
ELSE
LOCATE (40), 7
PRINT TAB(480); "                        "
LOCATE (41), 7
COLOR 13
PRINT TAB(480); a; "th TO "; TAB(480); a + 1; "th FLOOR";
TAB(480); "                  "
LOCATE (45), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Frc Cols/", TAB(480); "    Walls "; TAB(480); "    "
LOCATE (49), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT a + 1; "th Flr"; TAB(480); "    Frc Slab/",
TAB(480); " Beams"; TAB(480); "                  "
LOCATE (53), 1
PRINT TAB(480); "                              "
LOCATE (54), 1
```

```
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Frc Stair"; TAB(480); "            "
LOCATE (56), 1
PRINT TAB(480); "                         "
END IF
a = a + 1
cc = cc + 1
GOTO 156

IF a = nofloor THEN
GOTO 1222
END IF
LINE (0, 0)-(522, 0)
LINE (0, 432)-(522, 432)
GOTO 156

1222 CALL keyin(MINKEY(), MAXKEY(), EXTMINKEY(), EXTMAXKEY(),
SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON(), CALLER, XFLD,
YFLD, LNGTH, TRUE, FALSE, c$)




'**************ROOF*****************************************

1223 VIEW (100, 30)-(621, 479): CLS 1

IF ASC(c$) = 80 THEN
PAGE1 = PAGE1 + 1
END IF
LINE (0, 0)-(522, 0)
LINE (0, 432)-(522, 432)
IF bar = 1 THEN
FOR i = 0 TO 520 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF
90090 IF tt = 0 THEN
i = a + 3
CALL p2p12(qxxe(), qxx(), qyye(), qyy(), qzze(), qzz(), i, a)
END IF

1212 IF nofloor / 2 <> ww THEN
LOCATE (5), 1
PRINT , TAB(480); "            ";
LOCATE (9), 1
COLOR 13
PRINT , TAB(480); a - 1; "th TO "; TAB(480); " ROOF        "
LOCATE (13), 1
cc = cc - 2
COLOR 3
PRINT cc;
COLOR 15
```

```
PRINT "Frc Cols/", TAB(480); "    Walls      "; TAB(480); " "
LOCATE (17), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Roof          "; TAB(480); "    Frc Slab/",
TAB(480); "    Beam              "; TAB(480); "          "
LOCATE (21), 1
COLOR 12
PRINT , TAB(480); "BUILDING ", TAB(480); "ENVELOPE       "
LOCATE (24), 1
PRINT , TAB(480); "              "
LOCATE (25), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Internal    ", TAB(480); "    Walls          ",
TAB(480); "              "
LOCATE (28), 1
PRINT , TAB(480); "              "
LOCATE (29), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "External    ", TAB(480); "    Walls          ";
TAB(480); "                  "
LOCATE (32), 1
COLOR 12
PRINT , TAB(480); "ROOFING   ", TAB(480); "                ",
TAB(480); "                "
LOCATE (36), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Roof         "; TAB(480); "    Asphalt    ";
TAB(480); "              "
LOCATE (39), 1
cc = cc + 1
COLOR 12
PRINT "FINISHES          "
LOCATE (41), 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Internal     "; TAB(480); "              "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Wall             "; TAB(480); "              "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
```

```
PRINT "Floor            "; TAB(480); "              "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Ceiling         "; TAB(480); "              "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "External         "; TAB(480); "              "
COLOR 12
PRINT "SERVICES    "; TAB(480); "          "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Lifts           "; TAB(480); "              "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Mechanic.       "; TAB(480); "              "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Electric.       "; TAB(480); "              "
END IF

IF nofloor / 2 = ww THEN
LOCATE (5), 1
PRINT , TAB(480); "            ";
LOCATE (9), 1
COLOR 13
PRINT , TAB(480); a + 1; "th TO "; TAB(480); " ROOF       "
LOCATE (13), 1
cc = cc + 4
COLOR 3
PRINT cc;
COLOR 15

PRINT "Frc Cols/", TAB(480); "    Walls                 ";
TAB(480); "                "
LOCATE (17), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Roof           "; TAB(480); "    Frc Slab/",
TAB(480); "    Beams              "; TAB(480); "             "
LOCATE (21), 1
COLOR 12
PRINT , TAB(480); "BRICKWORK ", TAB(480); "BLOCKWORK        "
LOCATE (25), 1
PRINT , TAB(480); "               "
LOCATE (26), 1
cc = cc + 1
```

```
COLOR 3
PRINT cc;
COLOR 15
PRINT "Internal    ", TAB(480); "    Walls    "
LOCATE (28), 1
PRINT , TAB(480); "              ", TAB(480); "            ",
TAB(480); "                "
LOCATE (30), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT ; "External    ", TAB(480); "    Walls          "
LOCATE (32), 1
COLOR 12
PRINT , TAB(480); "ROOFING   ", TAB(480); "                "
LOCATE (35), 1
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Roof        "; TAB(480); "    Asphalt    "
LOCATE (39), 1
cc = cc + 1
COLOR 12
PRINT "FINISHES                  "
LOCATE (41), 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Internal     "; TAB(480); "             "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Wall            "; TAB(480); "             "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Floor           "; TAB(480); "             "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Ceiling         "; TAB(480); "             "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "External        "; TAB(480); "             "
COLOR 12
PRINT "SERVICES   "; TAB(480); "         "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Lifts           "; TAB(480); "             "
```

```
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Mechanic.       "; TAB(480); "                "
cc = cc + 1
COLOR 3
PRINT cc;
COLOR 15
PRINT "Electric.       "; TAB(480); "                "
END IF

yeni = 1
COLOR 15
VIEW (100, 30)-(621, 479): CLS 1
LINE (0, 0)-(522, 0)
LINE (0, 432)-(522, 432)
IF bar = 1 THEN
FOR i = 0 TO 520 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF

gig = gig + 1

IF gxxee(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t

jason = t + 1

DO WHILE axee(gig) >= 480 * jason
axee(gig) = axee(gig) - 480 * jason
jason = jason + 1
LOOP

axx(gig) = gxx1e(4) - 480 * t
ELSEIF gxxee(4) < 480 * t AND gxx1e(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t
axx(gig) = gxx1e(4) - 480 * t
ELSEIF gxx1e(4) < 480 * t THEN
axee(gig) = 0
axx(gig) = 0
END IF

gig1 = gig1 + 1

IF gxxeei(4) >= 480 * t THEN
axeei(gig1) = gxxeei(4) - 480 * t

jason1 = t + 1

DO WHILE axeei(gig1) >= 480 * jason1
axeei(gig1) = axeei(gig1) - 480 * jason1
jason1 = jason1 + 1
LOOP

axxi(gig1) = gxx1ei(4) - 480 * t
ELSEIF gxxeei(4) < 480 * t AND gxx1ei(4) >= 480 * t THEN
```

```
axeei(gig1) = gxxeei(4) - 480 * t
axxi(gig1) = gxx1ei(4) - 480 * t
ELSEIF gxx1ei(4) < 480 * t THEN
axeei(gig1) = 0
axxi(gig1) = 0
END IF

i = a + 2
COLOR 5
LINE (qxxe(i + 1), 80)-(qxx(i + 1), 85), , BF
LINE (qyye(i + 1), 110)-(qyy(i + 1), 115), , BF
COLOR 4
IF qyye(i + 1) = 0 THEN
LINE (0, 0)-(0, 0), , BF
ELSE
LINE (qyye(i + 1), 80)-(qyye(i + 1) + 1, 110), , BF
END IF
COLOR 15
IF gra1 < 0 THEN
GOTO 4452
END IF
IF qyye(i) = 0 THEN
grae = gyye(i) + lagras
ELSE
grae = qyye(i + 1) + lagras
END IF
gra1 = grae + gras

4452 COLOR 5

IF gra1 < 0 AND grae = 0 AND ASC(c$) = 77 THEN
LINE (0, 260)-(0, 265), , BF

ELSE
LINE (grae, 260)-(gra1, 265), , BF
        COLOR 10
        LINE (grae, 110)-(grae + 1, 260), , BF
END IF
COLOR 15
IF yeni = 0 THEN
IF i <= nof + 3 AND oh <> op THEN
CALL col(gxxee(), gxx1e(), fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)

ELSEIF i >= nof + 3 AND gig < i AND oh = 0 AND op = 0 THEN
CALL col(gxxee(), gxx1e(), fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)

ELSEIF i + 1 >= nof + 3 AND gig >= i AND oh <> op THEN
CALL col(gxxee(), gxx1e(), fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5)
```

```
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)

ELSEIF i >= nof + 3 AND oh > op AND gig > i THEN
CALL col(gxxee(), gxx1e(), fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
END IF

IF i <= nof1 + 3 AND oh <> op THEN
LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
IF poi > gzze(4) THEN
poi = gzze(4)
END IF
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 1
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5

ELSEIF i >= nof1 + 3 AND gig1 < i AND oh = 0 AND op = 0 THEN
LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF

IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
IF poi > gzze(4) THEN
poi = gzze(4)
END IF
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 1
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5

ELSEIF i + 1 >= nof1 + 3 AND gig1 >= i AND oh <> op THEN
LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
IF poi > gzze(4) THEN
poi = gzze(4)
END IF

DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 1
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
```

```
ELSEIF i >= nof1 + 3 AND oh > op AND gig1 > i THEN
LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
IF poi > gzze(4) THEN
poi = gzze(4)
END IF
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 1
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
END IF

ELSEIF yeni = 1 THEN
IF t = 0 AND gxxee(4) <= 480 THEN
LINE (gxxe(a + 3), 80)-(gxx1(a + 3), 85), , BF
LINE (gyye(a + 3), 110)-(gyy1(a + 3), 115), , BF
LINE (gyye(a + 3), 80)-(gyye(a + 3) + 1, 110), , BF
CALL col(gxxee(), gxx1e(), fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
ELSEIF t = 0 AND gxxee(4) > 480 THEN
LINE (0, 205)-(0, 210), , BF

ELSEIF oh = op AND t <> 0 THEN
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)

ELSEIF oh > op AND t <> 0 THEN
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, ser3, ser4, ser5,
ser6)
```

```
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)

ELSEIF oh < op THEN
CALL AL(c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin)
CALL sff(gxxee(), gxx1e(), fin1, FIN2, fin3, fin4, fin5,
fin6, fin7, fin8, fin9, fin10, ser1, ser2, aser3, aser4,
aser5, aser6)
CALL sf(afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i)
END IF
COLOR 5
IF t = 0 AND gxxeei(4) <= 480 THEN
LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF

IF gxxeei(4) < 480 THEN
poi = gxxeei(4)
IF poi < gzze(4) THEN
poi = gzze(4)
END IF
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5

ELSEIF t = 0 AND gxxeei(4) > 480 THEN
LINE (0, 175)-(0, 180), , BF

ELSEIF oh = op AND t <> 0 THEN
LINE (axeei(gig1), 175)-(axxi(gig1), 180), , BF

ELSEIF oh > op AND t <> 0 THEN
LINE (axeei(gig1), 175)-(axxi(gig1), 180), , BF

ELSEIF oh < op THEN
LINE (axeei(gig1), 175)-(axxi(gig1), 180), , BF
END IF
END IF
COLOR 15

IF nofloor / 2 <> ww AND a = nofloor OR nofloor / 2 = ww AND
a = nofloor - 2 THEN

CALL keyin(MINKEY(), MAXKEY(), EXTMINKEY(), EXTMAXKEY(),
SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON(), CALLER, XFLD,
YFLD, LNGTH, TRUE, FALSE, c$)
IF ASC(c$) = 27 THEN
```

```
CLOSE
GOTO 10000
END IF
IF page = 1 AND ASC(c$) = 75 AND t <> 0 THEN
GOTO 10
ELSEIF ASC(c$) = 75 THEN
t = t - 1
page = page
exc = 1
oh = oh + 1
tuf = tuf - 1
IF page = 0 OR PAGE1 < 0 OR t < 0 THEN
GOTO 10000
END IF
GOTO 77

ELSEIF ASC(c$) = 77 THEN
t = t
tuf = tuf + 1
che = 1
gig = gig + 1

IF gxxee(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t

jason = t + 1

DO WHILE axee(gig) >= 480 * jason
axee(gig) = axee(gig) - 480 * jason
jason = jason + 1
LOOP

axx(gig) = gxx1e(4) - 480 * t
ELSEIF gxxee(4) < 480 * t AND gxx1e(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t
axx(gig) = gxx1e(4) - 480 * t
ELSEIF gxx1e(4) < 480 * t THEN
axee(gig) = 0
axx(gig) = 0
END IF
gig1 = gig1 + 1

IF gxxeei(4) >= 480 * t THEN
axeei(gig1) = gxxeei(4) - 480 * t

jason1 = t + 1

DO WHILE axeei(gig1) >= 480 * jason1
axeei(gig1) = axeei(gig1) - 480 * jason1
jason1 = jason1 + 1
LOOP

axxi(gig1) = gxx1ei(4) - 480 * t
ELSEIF gxxeei(4) < 480 * t AND gxx1ei(4) >= 480 * t THEN
axeei(gig) = gxxeei(4) - 480 * t
axxi(gig) = gxx1ei(4) - 480 * t
ELSEIF gxx1ei(4) < 480 * t THEN
axeei(gig1) = 0
```

```
axxi(gig1) = 0
END IF


page = page + 1
exc = 1
op = op + 1
IF nofloor / 2 = ww AND a = nofloor - 2 THEN
df = 1
END IF
IF nofloor / 2 <> ww AND a = nofloor THEN
dff = 1
END IF
GOTO 777
ELSEIF ASC(c$) = 80 THEN
GOTO 1000
ELSEIF ASC(c$) = 72 THEN
PAGE1 = PAGE1 - 1
IF a < nofloor AND nofloor / 2 <> ww THEN
a = a - 4
cc = cc - 4
ELSEIF a < nofloor AND nofloor / 2 = ww THEN
a = a + 1
cc = cc - 11
ELSEIF a >= nofloor AND nofloor / 2 = ww THEN
a = a
cc = cc
ELSEIF a >= nofloor AND nofloor / 2 <> ww THEN
a = a
cc = cc + 8
END IF
IF PAGE1 < 0 OR a < 0 THEN
GOTO 10000
ELSEIF PAGE1 = 1 OR a = 1 THEN
GOTO 889
ELSEIF PAGE1 = 0 OR a = 1 THEN
GOTO 110
ELSEIF PAGE1 = 2 OR a = 2 THEN
GOTO 81
ELSEIF a > 2 THEN
GOTO 999
END IF
END IF
END IF


FOR i = 5 TO nofloor + 4
IF gxx1(i) >= 480 * t THEN
qxxe(i) = gxxe(i) - 480 * t
qxx(i) = gxx1(i) - 480 * t
ELSEIF gxx1(i) < 480 * t THEN
qxxe(i) = 0
qxx(i) = 0
END IF
IF gyy1(i) >= 480 * t THEN
qyye(i) = gyye(i) - 480 * t
qyy(i) = gyy1(i) - 480 * t
ELSEIF gyy1(i) < 480 * t THEN
```

```
qyye(i) = 0
qyy(i) = 0
END IF

IF gzze(i) >= 480 * t THEN
qzze(i) = gzz1(i) - 480 * t
qzz(i) = gzze(i) - 480 * t
ELSEIF gzze(i) < 480 * t THEN
qzze(i) = 0
qzz(i) = 0
END IF

IF i >= nof + 3 THEN
FOR gig = nof + 3 TO nofloor + 4
IF gxx1e(4) >= 480 * t THEN
axee(gig) = gxxee(4) - 480 * t
axx(gig) = gxx1e(4) - 480 * t
ELSEIF gxx1e(4) < 480 * t THEN
axee(gig) = 0
axx(gig) = 0
END IF
NEXT
END IF
CALL ngigi(fin1, FIN2, fin3, fin4, fin5, fin6, fin7, fin8,
fin9, fin10, ser1, ser2, ser3, ser4, ser5, ser6, afin1(),
afin2(), afin3(), afin4(), afin5(), afin6(), afin7(),
afin8(), afin9(), afin10(), aser1(), aser2(), aser3(),
aser4(), aser5(), aser6(), t, jf1, jf3, jf5, jf7, jf9, js1,
js3, js5, gig, nofloor)
IF i >= nof1 + 3 THEN
FOR gig1 = nof1 + 3 TO nofloor + 4
IF i(4) >= 480 * t THEN
axeei(gig1) = gxxeei(4) - 480 * t
axxi(gig1) = gxx1e(4) - 480 * t
ELSEIF gxx1ei(4) < 480 * t THEN
axeei(gig1) = 0
axxi(gig1) = 0
END IF
NEXT
END IF
NEXT

IF ASC(c$) = 75 OR ASC(c$) = 77 THEN
GOTO 1223
ELSEIF ASC(c$) = 80 THEN
GOTO 10000
ELSEIF ASC(c$) = 72 THEN
PAGE1 = PAGE1 - 1
a = a + 3
cc = cc + 3
ELSEIF PAGE1 = 2 OR a = 2 THEN
GOTO 81
ELSE
GOTO 9999
END IF

156 LINE (0, 0)-(522, 0)
LINE (0, 432)-(522, 432)
```

```
IF bar = 1 THEN
FOR i = 0 TO 520 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF
IF page = 1 AND PAGE1 = 2 THEN
CALL p1p12(gxxee(), gxxeei(), page, nof, nof1, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), hj1)
END IF
IF page = 1 AND PAGE1 > 2 AND a <= nofloor THEN
CALL p1p2g(gxxe(), gxx1(), gyye(),gyy1(), gzze(), gzz1(),i,a)
COLOR 5
IF nofloor / 2 = ww THEN
COLOR 5
LINE (gzze(i + 2), 410)-(gzz1(i + 2), 415), , BF
END IF
ELSEIF page >=2 AND PAGE1>2 AND a<=nofloor AND nofloor/2 <>ww
THEN
CALL p2p22(gxxee(), gxxeei(), nof, nof1, page, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), i, a)
IF nofloor / 2 = ww THEN
LINE (qzze(i + 3), 410)-(qzz1(i + 3), 415), , BF
END IF
IF a < nofloor THEN
LINE (qzze(i + 3), 410)-(qzz(i + 3), 415), , BF
END IF
END IF
COLOR 15

2345 CALL keyin(MINKEY(), MAXKEY(), EXTMINKEY(), EXTMAXKEY(),
SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON(), CALLER, XFLD,
YFLD, LNGTH, TRUE, FALSE, c$)
IF ASC(c$) = 27 THEN
CLOSE
GOTO 10000
END IF
IF page = 1 AND ASC(c$) = 75 AND t <> 0 THEN
GOTO 10
ELSEIF ASC(c$) = 75 THEN
t = t - 1
page = page - 1
'new'
IF PAGE1 >= nofloor - 4 THEN
tuf = tuf - 1
END IF
IF page = 0 OR PAGE1 < 0 OR t < 0 THEN
GOTO 10000
END IF
GOTO 77
ELSEIF ASC(c$) = 77 THEN
GOTO 777
ELSEIF ASC(c$) = 80 THEN
page = page
PAGE1 = PAGE1 + 1
IF PAGE1 = 2 THEN
GOTO 81
ELSEIF PAGE1 = 1 THEN
GOTO 888
```

```
ELSEIF PAGE1 > 2 THEN
GOTO 999
END IF
ELSEIF ASC(c$) = 72 THEN
PAGE1 = PAGE1 - 1
IF a < nofloor THEN
a = a - 4
cc = cc - 4
ELSEIF a >= nofloor THEN
a = a - 2
cc = cc - 2
END IF
IF PAGE1 < 0 OR a < 0 THEN
GOTO 10000
ELSEIF PAGE1 = 1 OR a = 1 THEN
GOTO 889
ELSEIF PAGE1 = 0 OR a = 1 THEN
GOTO 110
ELSEIF PAGE1 = 2 OR a = 2 THEN
GOTO 81
ELSEIF a > 2 THEN
GOTO 999
END IF
END IF

LINE (0, 0)-(522, 0)
LINE (0, 432)-(522, 432)
IF bar = 1 THEN
FOR i = 0 TO 520 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF
GOTO 10000

81 a = 2
VIEW (100, 30)-(621, 479): CLS 1
LOCATE 5, 1
PRINT , TAB(480); "                                    "
PRINT , TAB(480); "                                    "
cc = 15
FOR j = 5 TO 33 STEP 16
COLOR 13
LOCATE (4 + j), 1
PRINT , TAB(480); a; "th TO", TAB(480); a + 1; "th FLOOR";
TAB(480); "            "
COLOR 3
LOCATE (8 + j), 1
PRINT , TAB(480); cc;
COLOR 15
PRINT "Frc Cols/ "; TAB(480); "    Walls "; TAB(480); "     "
LOCATE (12 + j), 1
cc = cc + 1
COLOR 3
PRINT , TAB(480); cc;
COLOR 15
PRINT a + 1; "th Flr"; TAB(480); "    Frc Slab/",
TAB(480); "    Beams"; TAB(480); "             "; TAB(480);""
LOCATE (17 + j), 1
```

```
cc = cc + 1
COLOR 3
PRINT , TAB(480); cc;
COLOR 15
PRINT "Frc Stair"; TAB(480); "                    ";
TAB(480); "                "
cc = cc + 1
a = a + 1
NEXT
IF a = nofloor - 1 THEN
COLOR 13
LOCATE 41, 1
PRINT , TAB(480); a; "th TO", TAB(480); " ROOF          ";
TAB(480); "             "
COLOR 3
LOCATE 45, 1
PRINT , TAB(480); cc;
COLOR 15
PRINT "Frc Cols/ "; TAB(480); "    Walls ";
TAB(480); "                "
LOCATE 49, 1
cc = cc + 1
COLOR 3
PRINT , TAB(480); cc;
COLOR 15
PRINT a + 1; "th Flr"; TAB(480); "    Frc Slab/",
TAB(480); "    Beams"; TAB(480); "              "; TAB(480);""
LOCATE 54, 1
PRINT "                          "; TAB(480); "             "
LOCATE 56, 1
PRINT "                          "; TAB(480); "              "
cc = cc + 1
a = a + 1
ELSEIF a < nofloor - 1 THEN
COLOR 13
LOCATE 41, 1
PRINT , TAB(480); a; "th TO", TAB(480); a + 1; "th FLOOR  ";
TAB(480); "             "
COLOR 3
LOCATE 45, 1
PRINT , TAB(480); cc;
COLOR 15
PRINT "Frc Cols/ "; TAB(480); "    Walls "; TAB(480); "     "
LOCATE 49, 1
cc = cc + 1
COLOR 3
PRINT , TAB(480); cc;
COLOR 15
PRINT a + 1; "th Flr"; TAB(480); "    Frc Slab/",
TAB(480); "    Beams"; TAB(480); "              ";
TAB(480); "                "
LOCATE 54, 1
cc = cc + 1
COLOR 3
PRINT , TAB(480); cc;
COLOR 15
PRINT "Frc Stair"; TAB(480); "                    ";
TAB(480); "                "
```

```
LOCATE 56, 1
PRINT "                              "; TAB(480); "                "
cc = cc + 1
a = a + 1
END IF
LINE (0, 0)-(522, 0)
LINE (0, 432)-(522, 432)
IF bar = 1 THEN
FOR i = 0 TO 520 STEP 40
LINE (i, 0)-(i + 1, 432), , BF
NEXT
END IF
IF page = 1 AND PAGE1 = 2 AND a <= nofloor-1 OR nofloor/2=ww
THEN
CALL p1p12(gxxee(), gxxeei(), page, nof, nof1, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), hj1)

ELSEIF page = 1 AND PAGE1 = 2 AND a <= nofloor - 1 AND
nofloor / 2 <> ww THEN

CALL p1p2(gxxe(), gxx1(), gyye(), gyy1(), gzze(), gzz1(),
hj1)

ELSEIF page = 1 AND PAGE1 > 2 AND a < nofloor - 1 AND
nolfoor / 2 <> ww THEN

CALL p1p2g(gxxe(), gxx1(), gyye(), gyy1(), gzze(), gzz1(), i,
a)

ELSEIF page = 1 AND PAGE1 > 2 AND a <= nofloor - 1 AND
nolfoor / 2 = ww THEN

CALL p1p2g(gxxe(), gxx1(), gyye(), gyy1(), gzze(),gzz1(),i,a)

ELSEIF page >= 2 AND PAGE1 = 2 AND a < nofloor - 1 THEN
i = a + 1

CALL p2p12(qxxe(), qxx(), qyye(), qyy(), qzze(), qzz(), i, a)

ELSEIF page >= 2 AND PAGE1 = 2 AND a >= nofloor - 1 THEN

CALL p2p22(gxxee(), gxxeei(), nof, nof1, page, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), i, a)

END IF
CALL keyin(MINKEY(), MAXKEY(), EXTMINKEY(), EXTMAXKEY(),
SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON(), CALLER, XFLD,
YFLD, LNGTH, TRUE, FALSE, c$)
IF ASC(c$) = 27 THEN
CLOSE
GOTO 10000
END IF
IF a > nofloor AND ASC(c$) = 80 OR t < 0 THEN
GOTO 10000
ELSEIF ASC(c$) = 75 THEN
t = t - 1
page = page - 1
```

```
IF PAGE1 >= nofloor - 4 THEN
tuf = tuf - 1
END IF

IF page = 0 OR PAGE1 < 0 OR t < 0 THEN
GOTO 10000
END IF
GOTO 77

ELSEIF ASC(c$) = 77 THEN
GOTO 777
ELSEIF ASC(c$) = 80 THEN
PAGE1 = PAGE1 + 1
cc = cc - 3
GOTO 999
ELSEIF ASC(c$) = 72 THEN
PAGE1 = PAGE1 - 1
IF PAGE1 < 0 OR a < 0 THEN
GOTO 10000
ELSEIF PAGE1 = 1 THEN
GOTO 889
ELSEIF PAGE1 = 0 THEN
GOTO 110
ELSEIF PAGE1 = 1 AND page = 2 THEN
GOTO 889
ELSE
a = a - 4
IF a < 0 THEN
GOTO 10000
ELSE
GOTO 999
END IF
END IF
END IF

1000 IF ASC(c$) = 27 THEN
CLOSE
END IF
10000 CLOSE
END



SUB AL (c$, page, oh, op, i, nof, gig, gxxee(), gxx1e(),
fin1, fin3, fin5, fin7, fin9, ser1, ser3, ser5, axee(),
axx(), afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), uyu, tete, t, gig1, ece,
dede, yeni, gxxeei(), gxx1ei(), axeei(), axxi(), tete1,
murat1, uyu1, nof1, murat, kesin, yesin) STATIC

ohh = oh
opp = op
        COLOR 5
```

```
      IF i <= nof + 3 AND op <= oh AND opp <= ohh AND op = 0
AND gig = i AND murat = 0 AND tete = 1 THEN
      LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
murat = 1
    ELSEIF i >= nof + 3 AND oh > op AND op <> 0 AND ohh > opp
AND gig >= i AND murat = 0 AND tete = 1 THEN
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
LINE (po, 205)-(gxx1e(4), 210), , BF
CALL col1(gxxeei(), c$, age, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        tete = 1
     ELSEIF i >= nof + 3 AND oh = op AND ohh = opp AND op = 0
AND gig >= i AND tete = 0 THEN
LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
        tete = 1
CALL col1(gxxeei(), c$, age, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
    ELSEIF i <= nof + 3 AND oh = op AND op = 0 AND ohh = opp
AND gig > i AND t <> 0 AND murat = 0 AND tete = 0 THEN
        LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
    ELSEIF i >= nof + 3 AND oh = op AND oh <> 0 AND ohh = opp
AND gig >= i AND murat = 0 AND tete = 1 THEN
        LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
    ELSEIF i >= nof + 3 AND oh = op AND ohh = opp AND op = 0
AND gig >= i AND tete = 0 THEN
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
```

```
CALL col1(gxxeei(), c$, age, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
tete = 1
    ELSEIF i >= nof + 3 AND oh > op AND op <> 0 AND ohh > opp
AND gig >= i AND murat = 0 AND tete = 1 THEN
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        tete = 1
    ELSEIF i >= nof + 3 AND oh > op AND op = 0 AND ohh > opp
AND gig >= i AND uyu >= op AND murat = 0 AND tete = 0 AND t =
0 THEN
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())


ELSEIF i >= nof + 3 AND oh >= op AND ohh >= opp AND gig >= i
AND murat = 0 AND tete = 0 AND kesin < yesin AND t = 0 THEN
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof + 3 AND oh > op AND op = 0 AND ohh > opp
AND gig >= i AND kesin = yesin AND uyu > op AND murat = 0 AND
tete = 0 AND t = 0 THEN
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
```

```
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
    ELSEIF i >= nof + 3 AND oh > op AND op <> 0 AND ohh > opp
AND gig >= i AND kesin >= yesin AND murat = 0 AND tete = 0
AND t = 0 AND t = 0 THEN
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
ELSEIF i >= nof + 3 AND oh = op + 1 AND op <> 0 AND
ohh =opp + 1 AND gig >= i AND kesin = yesin - 1 AND murat = 0
AND tete = 0 AND t = 0 THEN
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
  ELSEIF i >= nof + 3 AND oh >= op AND op <> 0 AND ohh >= opp
AND gig >= i AND t = 0 AND murat = 0 AND tete = 0 THEN
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof + 3 AND oh > op AND ohh > opp AND gig >= i
AND murat = 0 AND tete = 1 AND t <> 0 THEN
        LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof + 3 AND oh > op AND op = 0 AND ohh > opp
AND gig >= i AND murat = 0 AND tete = 0 AND uyu <= op AND t
<> 0 THEN
```

```
        LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
CALL coll(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        uyu = uyu + 1
  ELSEIF i >= nof + 3 AND oh >= op AND op <> 0 AND ohh >= opp
AND gig >= i AND t <> 0 AND murat = 0 AND tete = 0 AND t <> 0
THEN
        LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
CALL coll(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof + 3 AND oh > op AND op <> 0 AND ohh > opp
AND gig >= i AND murat = 0 AND tete = 0 AND t <> 0 THEN
        LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
CALL coll(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof + 3 AND oh = op AND oh <> 0 AND ohh = opp
AND gig >= i AND murat = 0 AND tete = 1 THEN
        LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
CALL coll(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof + 3 AND op >= oh AND opp > ohh AND gig > i
THEN
        LINE (axee(i + 1), 205)-(axx(i + 1), 210), , BF
CALL coll(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof + 3 AND oh >= op AND opp < ohh AND gig < i
THEN
        LINE (axee(i - 1), 205)-(axx(i - 1), 210), , BF
        LINE (afin1(i - 1), 290)-(afin2(i - 1), 295), , BF
        LINE (afin3(i - 1), 308)-(afin4(i - 1), 313), , BF
        LINE (afin5(i - 1), 326)-(afin6(i - 1), 331), , BF
        LINE (afin7(i - 1), 344)-(afin8(i - 1), 349), , BF
        LINE (afin9(i - 1), 362)-(afin10(i - 1), 367), , BF
        LINE (aser1(i - 1), 390)-(aser2(i - 1), 395), , BF
        LINE (aser3(i - 1), 408)-(aser4(i - 1), 413), , BF
        LINE (aser5(i - 1), 424)-(aser6(i - 1), 429), , BF
CALL coll(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
END IF
        COLOR 5
    IF i <= nof1 + 3 AND op <= oh AND opp <= ohh AND op = 0
AND gig1 = i AND murat1 = 0 AND tete1 = 1 THEN
        LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
        murat1 = 1
CALL coll(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof1 + 3 AND oh = op AND ohh = opp AND op = 0
AND gig1 >= i AND tete1 = 0 THEN
        LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
```

```
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
        tete1 = 1
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof1 + 3 AND oh > op AND op <> 0 AND ohh > opp
AND gig1 >= i AND murat1 = 0 AND tete1 = 1 THEN
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
LINE (poi, 175)-(gxx1ei(4), 180), , BF
        tete1 = 1
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
    ELSEIF i >= nof1 + 3 AND oh > op AND op = 0 AND ohh > opp
AND gig1 >= i AND uyu1 >= op AND murat1 = 0 AND tete1 = 0 AND
t = 0 THEN
        LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
      ELSEIF i >= nof1 + 3 AND oh >= op AND ohh >= opp AND
gig1 >= i AND murat1 = 0 AND tete1 = 0 AND kesin < yesin AND
t = 0 THEN
        LINE (gxxeei(4), 175)-(gxx1e(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
```

```
    ELSEIF i >= nof1 + 3 AND oh > op AND op = 0 AND ohh > opp
AND gig1 >= i AND kesin = yesin AND uyu1 > op AND murat1 = 0
AND tete1 = 0 AND t = 0 THEN
        LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
   ELSEIF i >= nof1 + 3 AND oh > op AND op <> 0 AND ohh > opp
AND gig1 >= i AND kesin >= yesin AND murat1 = 0 AND tete1 = 0
AND t = 0 THEN
LINE (gxxeei(4), 175)-(gxx1e(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        ELSEIF i >= nof1 + 3 AND oh = op + 1 AND op <> 0 AND
ohh = opp + 1 AND gig1 >= i AND kesin = yesin - 1 AND murat1
= 0 AND tete1 = 0 AND t = 0 THEN
        LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
          ELSEIF i >= nof1 + 3 AND oh >= op AND op <> 0 AND
ohh >= opp AND gig1 >= i AND t = 0 AND murat1 = 0 AND tete1 =
0 THEN
        LINE (gxxeei(4), 175)-(gxx1ei(4), 180), , BF
IF gxxeei(4) > 480 THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
END IF
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
COLOR 5
```

```
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        ELSEIF i >= nof1 + 3 AND oh > op AND ohh > opp AND
gig1 >= i AND murat1 = 0 AND tete1 = 1 AND t <> 0 THEN
        LINE (axeei(i + 1), 175)-(axxi(i + 1), 180), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
    ELSEIF i >= nof1 + 3 AND oh > op AND op = 0 AND ohh > opp
AND gig1 >= i AND murat1 = 0 AND tete1 = 0 AND uyu1 <= op AND
t <> 0 THEN
        LINE (axeei(i + 1), 175)-(axxi(i + 1), 180), , BF
        uyu1 = uyu1 + 1
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
          ELSEIF i >= nof1 + 3 AND oh >= op AND op <> 0 AND
ohh >= opp AND gig1 >= i AND t <> 0 AND murat1 = 0 AND
tete1 = 0 AND t <> 0 THEN
        LINE (axeei(i + 1), 175)-(axxi(i + 1), 180), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
          ELSEIF i >= nof1 + 3 AND oh > op AND op <> 0 AND
ohh > opp AND gig1 >= i AND murat1 = 0 AND tete1 = 0 AND
t <> 0 THEN
        LINE (axeei(i + 1), 175)-(axxi(i + 1), 180), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        ELSEIF i >= nof1 + 3 AND oh = op AND oh <> 0 AND
ohh = opp AND gig1 >= i AND murat1 = 0 AND tete1 = 1 THEN
        LINE (axeei(i + 1), 175)-(axxi(i + 1), 180), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        ELSEIF i >= nof1 + 3 AND op >= oh AND opp > ohh AND
gig1 > i THEN
LINE (axeei(i + 1), 175)-(axxi(i + 1), 180), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        ELSEIF i >= nof1 + 3 AND oh >= op AND opp < ohh AND
gig1 < i THEN
        LINE (axeei(i - 1), 175)-(axxi(i - 1), 180), , BF
CALL col1(gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee())
        END IF
        COLOR 15
        END SUB




SUB col (gxxee(), gxx1e(), fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5) STATIC
```

```
COLOR 5
LINE (gxxee(4), 205)-(gxx1e(4), 210), , BF
IF gxxee(4) > 480 THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
END IF
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
        COLOR 4
        LINE (fin1, 205)-(fin1 + 1, 290), , BF
        LINE (fin3, 205)-(fin3 + 1, 308), , BF
        LINE (fin5, 205)-(fin5 + 1, 326), , BF
        LINE (fin7, 205)-(fin7 + 1, 344), , BF
        LINE (fin9, 205)-(fin9 + 1, 362), , BF
        LINE (ser1, 205)-(ser1 + 1, 390), , BF
        LINE (ser3, 205)-(ser3 + 1, 408), , BF
        LINE (ser5, 205)-(ser5 + 1, 424), , BF
COLOR 15

END SUB
```

```
SUB col1 (gxxeei(), c$, page, fin1, fin3, fin5, fin7, fin9,
ser1, ser3, ser5, gxxee()) STATIC
```

```
IF gxxee(4) > 480 * (page - 2) AND gxxee(4) < 480 * (page -1)
THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
COLOR 14
LINE (po, 195)-(po + 2, 205), , BF
END IF
COLOR 5
IF gxxeei(4) > 480 * (page - 2) AND gxxeei(4) < 480 *(page-1)
THEN
poi = gxxeei(4)
DO UNTIL poi < 480
poi = poi - 480
LOOP
COLOR 9
LINE (poi, 165)-(poi + 2, 180), , BF
END IF
        IF fin1 > gxxee(4) + 480 * (M1 + 1) THEN
        LINE (0, 0)-(0, 0)
  ELSEIF F1 = 1 AND fin1 < 480 * page AND ASC(c$) <> 75 THEN
        LINE (0, 0)-(0, 0)
ELSEIF fin1 <= 480 *(page-1) AND fin1 >= 480 * (page -2)THEN
        IF ff1 = 0 OR ASC(c$) = 75 THEN
        ff1 = fin1
        IF ff1 > 480 THEN
        DO UNTIL ff1 < 480
        ff1 = ff1 - 480
        LOOP
        END IF
        LINE (ff1, 205)-(ff1 + 1, 290), , BF
        END IF
        ELSE
        ff1 = fin1
        IF ff1 > 480 THEN
        DO UNTIL ff1 < 480
        ff1 = ff1 - 480
        LOOP
IF fin1 <= 480 * (page - 1) AND fin1 >= 480 * (page - 2) THEN
        LINE (ff1, 205)-(ff1 + 1, 290), , BF
        F1 = 1
ELSEIF fin1 <= 480 * (page) AND fin1 >= 480 * (page - 1) AND
ASC(c$) = 80 THEN
        LINE (ff1, 205)-(ff1 + 1, 290), , BF
        END IF
        ELSE
        LINE (0, 0)-(0, 0)
        END IF
        END IF
        IF PAGE1 = page THEN
        M1 = M1 + 1
        END IF
        PAGE1 = page + 1
        IF fin3 > gxxee(4) + 480 * (M3 + 1) THEN
        LINE (0, 0)-(0, 0)
ELSEIF F3 = 1 AND fin3 < 480 * page AND ASC(c$) <> 75 THEN
```

```
        LINE (0, 0)-(0, 0)
ELSEIF fin3 <= 480 * (page - 1) AND fin3 >= 480 * (page - 2)
THEN
        IF ff3 = 0 OR ASC(c$) = 75 THEN
        ff3 = fin3
        IF ff3 > 480 THEN
        DO UNTIL ff3 < 480
        ff3 = ff3 - 480
        LOOP
        END IF
        LINE (ff3, 205)-(ff3 + 1, 318), , BF
        END IF
        ELSE
        ff3 = fin3
        IF ff3 > 480 THEN
        DO UNTIL ff3 < 480
        ff3 = ff3 - 480
        LOOP
IF fin3 <= 480 * (page - 1) AND fin3 >= 480 * (page - 2)
AND ff3 <> 0 THEN
        LINE (ff3, 205)-(ff3 + 1, 318), , BF
        F3 = 1
ELSEIF fin3 <= 480 * (page) AND fin3 >= 480 * (page - 1) AND
ASC(c$) = 80 THEN
        LINE (ff3, 205)-(ff3 + 1, 318), , BF
        END IF
        ELSE
        LINE (0, 0)-(0, 0)
        END IF
        END IF
        IF page3 = page THEN
        M3 = M3 + 1
        END IF
        page3 = page + 1
        IF fin5 > gxxee(4) + 480 * (M5 + 1) THEN
        LINE (0, 0)-(0, 0)
ELSEIF F5 = 1 AND fin5 < 480 * page AND ASC(c$) <> 75 THEN
        LINE (0, 0)-(0, 0)
ELSEIF fin5 <= 480 * (page - 1) AND fin5 >= 480 * (page - 2)
THEN
        IF ff5 = 0 OR ASC(c$) = 75 THEN
        ff5 = fin5
        IF ff5 > 480 THEN
        DO UNTIL ff5 < 480
        ff5 = ff5 - 480
        LOOP
        END IF
        LINE (ff5, 205)-(ff5 + 1, 326), , BF
        END IF
        ELSE
        ff5 = fin5
        IF ff5 > 480 THEN
        DO UNTIL ff5 < 480
        ff5 = ff5 - 480
        LOOP
IF fin5 <= 480 * (page - 1) AND fin5 >= 480 * (page - 2) THEN
        LINE (ff5, 205)-(ff5 + 1, 326), , BF
        F5 = 1
```

```
   ELSEIF fin5 <= 480 * (page) AND fin5 >= 480 * (page - 1)
AND ASC(c$) = 80 THEN
        LINE (ff5, 205)-(ff5 + 1, 326), , BF
        END IF
        ELSE
        LINE (0, 0)-(0, 0)
        END IF
        END IF
        IF page5 = page THEN
        M5 = M5 + 1
        END IF
        page5 = page + 1
        IF fin7 > gxxee(4) + 480 * (M7 + 1) THEN
        LINE (0, 0)-(0, 0), , BF
 ELSEIF F7 = 1 AND fin7 < 480 * page AND ASC(c$) <> 75 THEN
        LINE (0, 0)-(0, 0)
ELSEIF fin7 <= 480 * (page - 1) AND fin7 >= 480 * (page - 2)
THEN
        IF ff7 = 0 OR ASC(c$) = 75 THEN
        ff7 = fin7
        IF ff7 > 480 THEN
        DO UNTIL ff7 < 480
        ff7 = ff7 - 480
        LOOP
        END IF
        LINE (ff7, 205)-(ff7 + 1, 344), , BF
        END IF
        ELSE
        ff7 = fin7
        IF ff7 > 480 THEN
        DO UNTIL ff7 < 480
        ff7 = ff7 - 480
        LOOP
IF fin7 <= 480 * (page - 1) AND fin7 >= 480 * (page - 2) THEN
        LINE (ff7, 205)-(ff7 + 1, 344), , BF
        F7 = 1
ELSEIF fin7 <= 480 * (page) AND fin7 >= 480 * (page - 1) AND
ASC(c$) = 80 THEN
        LINE (ff7, 205)-(ff7 + 1, 344), , BF
        END IF
        ELSE
        LINE (0, 0)-(0, 0)
        END IF
        END IF
        IF page7 = page THEN
        M7 = M7 + 1
        END IF
        page7 = page + 1
        IF fin9 > gxxee(4) + 480 * (M9 + 1) THEN
        LINE (0, 0)-(0, 0)
ELSEIF F9 = 1 AND fin9 < 480 * page AND ASC(c$) <> 75 THEN
        LINE (0, 0)-(0, 0)
ELSEIF fin9 <= 480 * (page - 1) AND fin9 >= 480 * (page - 2)
THEN
        IF ff9 = 0 OR ASC(c$) = 75 THEN
        ff9 = fin9
        IF ff9 > 480 THEN
        DO UNTIL ff9 < 480
```

```
        ff9 = ff9 - 480
        LOOP
        END IF
        LINE (ff9, 205)-(ff9 + 1, 362), , BF
        END IF
 ELSE
        ff9 = fin9
        IF ff9 > 480 THEN
        DO UNTIL ff9 < 480
        ff9 = ff9 - 480
        LOOP
IF fin9 <= 480 * (page - 1) AND fin9 >= 480 * (page - 2) THEN
        LINE (ff9, 205)-(ff9 + 1, 362), , BF
        F9 = 1
ELSEIF fin9 <= 480 * (page) AND fin9 >= 480 * (page - 1) AND
ASC(c$) = 80 THEN
        LINE (ff9, 205)-(ff9 + 1, 362), , BF
        END IF
        ELSE
        LINE (0, 0)-(0, 0)
        END IF
        END IF
        IF page9 = page THEN
        M9 = M9 + 1
        END IF
        page9 = page + 1
        IF ser1 > gxxee(4) + 480 * (MS1 + 1) THEN
        LINE (0, 0)-(0, 0)
ELSEIF S1 = 1 AND ser1 < 480 * page AND ASC(c$) <> 75 THEN
        LINE (0, 0)-(0, 0)
ELSEIF ser1 <= 480 * (page - 1) AND ser1 >= 480 * (page - 2)
THEN
        IF ss1 = 0 OR ASC(c$) = 75 THEN
        ss1 = ser1
        IF ss1 > 480 THEN
        DO UNTIL ss1 < 480
        ss1 = ss1 - 480
        LOOP
        END IF
        LINE (ss1, 205)-(ss1 + 1, 390), , BF
        END IF
        ELSE
        ss1 = ser1
        IF ss1 > 480 THEN
        DO UNTIL ss1 < 480
        ss1 = ss1 - 480
        LOOP
IF ser1 <= 480 * (page - 1) AND ser1 >= 480 * (page - 2) THEN
        LINE (ss1, 205)-(ss1 + 1, 390), , BF
        S1 = 1
ELSEIF ser1 <= 480 * (page) AND ser1 >= 480 * (page - 1) AND
ASC(c$) = 80 THEN
        LINE (ss1, 205)-(ss1 + 1, 390), , BF
        END IF
        ELSE
        LINE (0, 0)-(0, 0)
        END IF
        END IF
```

```
        IF pags1 = page THEN
        MS1 = MS1 + 1
        END IF
        pags1 = page + 1
        IF ser3 > gxxee(4) + 480 * (MS3 + 1) THEN
        LINE (0, 0)-(0, 0)
ELSEIF S3 = 1 AND ser3 < 480 * page AND ASC(c$) <> 75 THEN
        LINE (0, 0)-(0, 0)
ELSEIF ser3 <= 480 * (page - 1) AND ser3 >= 480 * (page - 2)
THEN
        IF ss3 = 0 OR ASC(c$) = 75 THEN
        ss3 = ser3
        IF ss3 > 480 THEN
        DO UNTIL ss3 < 480
        ss3 = ss3 - 480
        LOOP
        END IF
        LINE (ss3, 205)-(ss3 + 1, 408), , BF
        END IF
        ELSE
        ss3 = ser3
        IF ss3 > 480 THEN
        DO UNTIL ss3 < 480
        ss3 = ss3 - 480
        LOOP
IF ser3 <= 480 * (page - 1) AND ser3 >= 480 * (page - 2) THEN
        LINE (ss3, 205)-(ss3 + 1, 408), , BF
        S3 = 1
ELSEIF ser3 <= 480 * (page) AND ser3 >= 480 * (page - 1) AND
ASC(c$) = 80 THEN
        LINE (ss3, 205)-(ss3 + 1, 408), , BF
        END IF
        ELSE
        LINE (0, 0)-(0, 0)
        END IF
        END IF
        IF pags3 = page THEN
        MS3 = MS3 + 1
        END IF
        pags3 = page + 1
        IF ser5 > gxxee(4) + 480 * (MS5 + 1) THEN
        LINE (0, 0)-(0, 0)
ELSEIF S5 = 1 AND ser5 < 480 * page AND ASC(c$) <> 75 THEN
        LINE (0, 0)-(0, 0)
ELSEIF ser5 <= 480 * (page - 1) AND ser5 >= 480 * (page - 2)
THEN
        IF ss5 = 0 OR ASC(c$) = 75 THEN
        ss5 = ser5
        IF ss5 > 480 THEN
        DO UNTIL ss5 < 480
        ss5 = ss5 - 480
        LOOP
        END IF
        LINE (ss5, 205)-(ss5 + 1, 424), , BF
        END IF
        ELSE
        ss5 = ser5
        IF ss5 > 480 THEN
```

```
        DO UNTIL ss5 < 480
        ss5 = ss5 - 480
        LOOP
IF ser5 <= 480 * (page - 1) AND ser5 >= 480 * (page - 2) THEN
        LINE (ss5, 205)-(ss5 + 1, 424), , BF
        S5 = 1
ELSEIF ser5 <= 480 * (page) AND ser5 >= 480 * (page - 1) AND
ASC(c$) = 80 THEN
        LINE (ss5, 205)-(ss5 + 1, 424), , BF
        END IF
        ELSE
        LINE (0, 0)-(0, 0)
        END IF
        END IF
        IF pags5 = page THEN
        MS5 = MS5 + 1
        END IF
        pags5 = page + 1
END SUB
```

```
SUB gigi (fin1, FIN2, fin3, fin4, fin5, fin6, fin7, fin8,
fin9, fin10, ser1, ser2, ser3, ser4, ser5, ser6, afin1(),
afin2(), afin3(), afin4(), afin5(), afin6(), afin7(),
afin8(), afin9(), afin10(), aser1(), aser2(), aser3(),
aser4(), aser5(), aser6(), t, jf1, jf3, jf5, jf7, jf9, js1,
js3, js5, gig) STATIC

IF fin1 >= 480 * t THEN
afin1(gig) = fin1 - 480 * t
jf1 = t + 1
DO WHILE afin1(gig) >= 480 * jf1
afin1(gig) = afin1(gig) - 480 * jf1
jf1 = jf1 + 1
LOOP
afin2(gig) = FIN2 - 480 * t

ELSEIF fin1 < 480 * t AND FIN2 >= 480 * t THEN
afin1(gig) = fin1 - 480 * t
afin2(gig) = FIN2 - 480 * t
ELSEIF FIN2 < 480 * t THEN
afin1(gig) = 0
afin2(gig) = 0
END IF

IF fin3 >= 480 * t THEN
afin3(gig) = fin3 - 480 * t
jf3 = t + 1
DO WHILE afin3(gig) >= 480 * jf3
afin3(gig) = afin3(gig) - 480 * jf3
jf3 = jf3 + 1
LOOP
afin4(gig) = fin4 - 480 * t
ELSEIF fin3 < 480 * t AND fin4 >= 480 * t THEN
afin3(gig) = fin3 - 480 * t
afin4(gig) = fin4 - 480 * t
ELSEIF fin4 < 480 * t THEN
afin3(gig) = 0
afin4(gig) = 0
END IF
IF fin5 >= 480 * t THEN
afin5(gig) = fin5 - 480 * t
jf5 = t + 1
DO WHILE afin5(gig) >= 480 * jf5
afin5(gig) = afin5(gig) - 480 * jf5
jf5 = jf5 + 1
LOOP
afin6(gig) = fin6 - 480 * t
ELSEIF fin5 < 480 * t AND fin6 >= 480 * t THEN
afin5(gig) = fin5 - 480 * t
afin6(gig) = fin6 - 480 * t
ELSEIF fin6 < 480 * t THEN
afin5(gig) = 0
afin6(gig) = 0
END IF
IF fin7 >= 480 * t THEN
```

```
afin7(gig) = fin7 - 480 * t
jf7 = t + 1
DO WHILE afin7(gig) >= 480 * jf7
afin7(gig) = afin7(gig) - 480 * jf7
jf7 = jf7 + 1
LOOP
afin8(gig) = fin8 - 480 * t
ELSEIF fin7 < 480 * t AND fin8 >= 480 * t THEN
afin7(gig) = fin7 - 480 * t
afin8(gig) = fin8 - 480 * t
ELSEIF fin8 < 480 * t THEN
afin7(gig) = 0
afin8(gig) = 0
END IF
IF fin9 >= 480 * t THEN
afin9(gig) = fin9 - 480 * t
jf9 = t + 1
DO WHILE afin9(gig) >= 480 * jf9
afin9(gig) = afin9(gig) - 480 * jf9
jf9 = jf9 + 1
LOOP
afin10(gig) = fin10 - 480 * t
ELSEIF fin9 < 480 * t AND fin10 >= 480 * t THEN
afin9(gig) = fin9 - 480 * t
afin10(gig) = fin10 - 480 * t
ELSEIF fin10 < 480 * t THEN
afin9(gig) = 0
afin10(gig) = 0
END IF
IF ser1 >= 480 * t THEN
aser1(gig) = ser1 - 480 * t
js1 = t + 1
DO WHILE aser1(gig) >= 480 * js1
aser1(gig) = aser1(gig) - 480 * js1
js1 = js1 + 1
LOOP
aser2(gig) = ser2 - 480 * t
ELSEIF ser1 < 480 * t AND ser2 >= 480 * t THEN
aser1(gig) = ser1 - 480 * t
aser2(gig) = ser2 - 480 * t
ELSEIF ser2 < 480 * t THEN
aser1(gig) = 0
aser2(gig) = 0
END IF
IF ser3 >= 480 * t THEN
aser3(gig) = ser3 - 480 * t
js3 = t + 1
DO WHILE aser3(gig) >= 480 * js3
aser3(gig) = aser3(gig) - 480 * js3
js3 = js3 + 1
LOOP
aser4(gig) = ser4 - 480 * t
ELSEIF ser3 < 480 * t AND ser4 >= 480 * t THEN
aser3(gig) = ser3 - 480 * t
aser4(gig) = ser4 - 480 * t
ELSEIF ser4 < 480 * t THEN
aser3(gig) = 0
aser4(gig) = 0
```

```
END IF
IF ser5 >= 480 * t THEN
aser5(gig) = ser5 - 480 * t
js5 = t + 1
DO WHILE aser5(gig) >= 480 * js5
aser5(gig) = aser5(gig) - 480 * js5
js5 = js5 + 1
LOOP
aser6(gig) = ser6 - 480 * t
ELSEIF ser5 < 480 * t AND ser6 >= 480 * t THEN
aser5(gig) = ser5 - 480 * t
aser6(gig) = ser6 - 480 * t
ELSEIF ser6 < 480 * t THEN
aser5(gig) = 0
aser6(gig) = 0
END IF
END SUB
```

```
SUB gullu (aemel)
IF bjason = 0 THEN
LINE (0, 400)-(aemel, 405)
END IF
END SUB
```

```
SUB keyin (MINKEY(), MAXKEY(), EXTMINKEY(), EXTMAXKEY(),
SPECIALKEY$(), EXTSPECIALKEY$(), CAPSON(), CALLER, XFLD,
YFLD, LNGTH, TRUE, FALSE, c$)

26115 c$ = INKEY$
IF c$ = "" THEN 26115: 'WAIT FOR INPUT
c = ASC(c$)
IF LEN(c$) = 2 THEN EXTENDED = TRUE ELSE EXTENDED = FALSE
IF EXTENDED = FALSE THEN 26155
c$ = RIGHT$(c$, 1)
c = ASC(c$)
GOTO 26205
26155 'ORDINARY ASCII CODES
'TEST FOR RANGE
IF c >= MINKEY(CALLER) AND c <= MAXKEY(CALLER) THEN 26185
'HANDLE SPECIAL CHARACTERS
IF SPECIALKEY$(CALLER) = " " THEN 26240: 'NO
IF INSTR(SPECIALKEY$(CALLER), c$) = 0 THEN 26240
26185 'CONVERT CAPITALS IF NECESSARY
IF CAPSON(CALLER) = FALSE THEN 26255
IF c > 96 AND c < 123 THEN c$ = CHR$(c AND 223)
GOTO 26255
26205 'EXTENDED ASCIII CODES
'TEST FOR RANGE
IF c >= EXTMINKEY(CALLER) AND c <= EXTMINKEY(CALLER) THEN
26255
'HANDLE SPECIAL CHARACTERS
IF EXTSPECIALKEY$(CALLER) = " " THEN 26240
IF INSTR(EXTSPECIALKEY$(CALLER), c$) = 1 THEN 26240
GOTO 26255
IF ASC(c$) = 81 THEN 26240
IF ASC(c$) = 82 THEN 26240
IF ASC(c$) = 83 THEN 26240
IF ASC(c$) = 60 THEN 26240
IF ASC(c$) = 61 THEN 26240
IF ASC(c$) = 63 THEN 26240
IF ASC(c$) = 64 THEN 26240
IF ASC(c$) = 65 THEN 26240
IF ASC(c$) = 66 THEN 26240
IF ASC(c$) = 67 THEN 26240

IF ASC(c$) = 73 THEN 26240
IF ASC(c$) = 79 THEN 26240
IF ASC(c$) = 71 THEN 26240
GOTO 26255
26240 'ILLEGAL CHARACTER
BEEP
GOTO 26115: 'TRY AGAIN
26255 END SUB
```

```
SUB ngigi (fin1, FIN2, fin3, fin4, fin5, fin6, fin7, fin8,
fin9, fin10, ser1, ser2, ser3, ser4, ser5, ser6, afin1(),
afin2(), afin3(), afin4(), afin5(), afin6(), afin7(),
afin8(), afin9(), afin10(), aser1(), aser2(), aser3(),
aser4(), aser5(), aser6(), t, jf1, jf3, jf5, jf7, jf9, js1,
js3, js5, gig, nofloor) STATIC

FOR gig = nof + 3 TO nofloor + 4
IF fin1 >= 480 * t THEN
afin1(gig) = fin1 - 480 * t
jf1 = t + 1
DO WHILE afin1(gig) >= 480 * jf1
afin1(gig) = afin1(gig) - 480 * jf1
jf1 = jf1 + 1
LOOP
afin2(gig) = FIN2 - 480 * t
ELSEIF fin1 < 480 * t AND FIN2 >= 480 * t THEN
afin1(gig) = fin1 - 480 * t
afin2(gig) = FIN2 - 480 * t
ELSEIF FIN2 < 480 * t THEN
afin1(gig) = 0
afin2(gig) = 0
END IF
NEXT
FOR gig = nof + 3 TO nofloor + 4
IF fin3 >= 480 * t THEN
afin3(gig) = fin3 - 480 * t
jf3 = t + 1
DO WHILE afin3(gig) >= 480 * jf3
afin3(gig) = afin3(gig) - 480 * jf3
jf3 = jf3 + 1
LOOP
afin4(gig) = fin4 - 480 * t
ELSEIF fin3 < 480 * t AND fin4 >= 480 * t THEN
afin3(gig) = fin3 - 480 * t
afin4(gig) = fin4 - 480 * t
ELSEIF fin4 < 480 * t THEN
afin3(gig) = 0
afin4(gig) = 0
END IF
NEXT
FOR gig = nof + 3 TO nofloor + 4
IF fin5 >= 480 * t THEN
afin5(gig) = fin5 - 480 * t
jf5 = t + 1
DO WHILE afin5(gig) >= 480 * jf5
afin5(gig) = afin5(gig) - 480 * jf5
jf5 = jf5 + 1
LOOP
afin6(gig) = fin6 - 480 * t
```

```
ELSEIF fin5 < 480 * t AND fin6 >= 480 * t THEN
afin5(gig) = fin5 - 480 * t
afin6(gig) = fin6 - 480 * t
ELSEIF fin6 < 480 * t THEN
afin5(gig) = 0
afin6(gig) = 0
END IF
NEXT

FOR gig = nof + 3 TO nofloor + 4
IF fin7 >= 480 * t THEN
afin7(gig) = fin7 - 480 * t
jf7 = t + 1
DO WHILE afin7(gig) >= 480 * jf7
afin7(gig) = afin7(gig) - 480 * jf7
jf7 = jf7 + 1
LOOP
afin8(gig) = fin8 - 480 * t
ELSEIF fin7 < 480 * t AND fin8 >= 480 * t THEN
afin7(gig) = fin7 - 480 * t
afin8(gig) = fin8 - 480 * t
ELSEIF fin8 < 480 * t THEN
afin7(gig) = 0
afin8(gig) = 0
END IF
NEXT
FOR gig = nof + 3 TO nofloor + 4
IF fin9 >= 480 * t THEN
afin9(gig) = fin9 - 480 * t
jf9 = t + 1
DO WHILE afin9(gig) >= 480 * jf9
afin9(gig) = afin9(gig) - 480 * jf9
jf9 = jf9 + 1
LOOP
afin10(gig) = fin10 - 480 * t
ELSEIF fin9 < 480 * t AND fin10 >= 480 * t THEN
afin9(gig) = fin9 - 480 * t
afin10(gig) = fin10 - 480 * t
ELSEIF fin10 < 480 * t THEN
afin9(gig) = 0
afin10(gig) = 0
END IF
NEXT
FOR gig = nof + 3 TO nofloor + 4
IF ser1 >= 480 * t THEN
aser1(gig) = ser1 - 480 * t
js1 = t + 1
DO WHILE aser1(gig) >= 480 * js1
aser1(gig) = aser1(gig) - 480 * js1
js1 = js1 + 1
LOOP
aser2(gig) = ser2 - 480 * t
ELSEIF ser1 < 480 * t AND ser2 >= 480 * t THEN
aser1(gig) = ser1 - 480 * t
aser2(gig) = ser2 - 480 * t
ELSEIF ser2 < 480 * t THEN
aser1(gig) = 0
aser2(gig) = 0
```

```
END IF
NEXT
FOR gig = nof + 3 TO nofloor + 4
IF ser3 >= 480 * t THEN
aser3(gig) = ser3 - 480 * t
js3 = t + 1
DO WHILE aser3(gig) >= 480 * js3
aser3(gig) = aser3(gig) - 480 * js3
js3 = js3 + 1
LOOP
aser4(gig) = ser4 - 480 * t
ELSEIF ser3 < 480 * t AND ser4 >= 480 * t THEN
aser3(gig) = ser3 - 480 * t
aser4(gig) = ser4 - 480 * t
ELSEIF ser4 < 480 * t THEN
aser3(gig) = 0
aser4(gig) = 0
END IF
NEXT
FOR gig = nof + 3 TO nofloor + 4
IF ser5 >= 480 * t THEN
aser5(gig) = ser5 - 480 * t
js5 = t + 1
DO WHILE aser5(gig) >= 480 * js5
aser5(gig) = aser5(gig) - 480 * js5
js5 = js5 + 1
LOOP
aser6(gig) = ser6 - 480 * t
ELSEIF ser5 < 480 * t AND ser6 >= 480 * t THEN
aser5(gig) = ser5 - 480 * t
aser6(gig) = ser6 - 480 * t
ELSEIF ser6 < 480 * t THEN
aser5(gig) = 0
aser6(gig) = 0
END IF
NEXT
END SUB
```

```
SUB p1p10 (g1xxe, g1xx1, g1yye, g1yy1, g1zze, g1zz1, g1tte,
g1tt1, g1uue, g1uu1, g22b, g22b1, g2xxe, g2xx1, g2yye, g2yy1,
g2zze, g2zz1, gyye(), gyy1(), gxxe(), gxx1()) STATIC
COLOR 5
LINE (g1xxe, 40)-(g1xx1, 45), , BF
LINE (g1yye, 70)-(g1yy1, 75), , BF
LINE (g1zze, 100)-(g1zz1, 105), , BF
LINE (g1tte, 100)-(g1tt1, 105), , BF
LINE (g1uue, 100)-(g1uu1, 105), , BF
LINE (g22b, 150)-(g22b1, 155), , BF
LINE (g2xxe, 180)-(g2xx1, 185), , BF
LINE (g2yye, 210)-(g2yy1, 215), , BF
LINE (g2zze, 250)-(g2zz1, 255), , BF
LINE (gyye(44), 315)-(gyy1(44), 320), , BF
LINE (gxxe(4), 400)-(gxx1(4), 405), , BF
COLOR 4
LINE (g1yye, 40)-(g1yye + 1, 70), , BF
LINE (g1zze, 70)-(g1zze + 1, 100), , BF
LINE (g22b, 100)-(g22b + 1, 150), , BF
LINE (g2xxe, 150)-(g2xxe + 1, 180), , BF
LINE (g2yye, 180)-(g2yye + 1, 210), , BF
LINE (g2zze, 210)-(g2zze + 1, 250), , BF
LINE (gyye(44), 250)-(gyye(44) + 1, 315), , BF
LINE (gxxe(4), 315)-(gxxe(4) + 1, 400), , BF
COLOR 15
END SUB
```

```
SUB p1p10e (xxe, xx, yye, YY, zze, zz, tte, tt, uue, uu,
xxxe2, xxx2, xxe2, xx2, yye2, yy2, zze2, zz2, qyye(), qyy1(),
qxxe(), qxx())
COLOR 5
LINE (xxe, 40)-(xx, 45), , BF
LINE (yye, 70)-(YY, 75), , BF
LINE (zze, 100)-(zz, 105), , BF
LINE (tte, 100)-(tt, 105), , BF
LINE (uue, 100)-(uu, 105), , BF
LINE (xxxe2, 150)-(xxx2, 155), , BF
LINE (xxe2, 180)-(xx2, 185), , BF
LINE (yye2, 210)-(yy2, 215), , BF
LINE (zze2, 250)-(zz2, 255), , BF
LINE (qyye(44), 315)-(qyy1(44), 320), , BF
LINE (qxxe(4), 400)-(qxx(4), 405), , BF
COLOR 4
LINE (yye, 40)-(yye + 1, 70), , BF
LINE (zze, 70)-(zze + 1, 100), , BF
LINE (xxxe2, 100)-(xxxe2 + 1, 150), , BF
LINE (xxe2, 150)-(xxe2 + 1, 180), , BF
LINE (yye2, 180)-(yye2 + 1, 210), , BF
LINE (zze2, 210)-(zze2 + 1, 250), , BF
LINE (qyye(44), 250)-(qyye(44) + 1, 315), , BF
LINE (qxxe(4), 315)-(qxxe(4) + 1, 400), , BF

COLOR 15

END SUB
```

```
SUB p1p11 (gxxee(), gxxeei(), nof, nof1, gxxe(), gxx1(),
gyye(), gyy1(), gzze(), gzz1(), hj) STATIC
page = 1
hj = 4
COLOR 5
LINE (gxxe(hj), 80)-(gxx1(hj), 85), , BF
LINE (gyye(hj), 110)-(gyy1(hj), 115), , BF
LINE (gxxe(hj + 1), 210)-(gxx1(hj + 1), 215), , BF
LINE (gyye(hj + 1), 240)-(gyy1(hj + 1), 245), , BF
LINE (gxxe(hj + 2), 340)-(gxx1(hj + 2), 345), , BF
LINE (gyye(hj + 2), 370)-(gyy1(hj + 2), 375), , BF
COLOR 4
LINE (gyye(hj), 80)-(gyye(hj) + 1, 110), , BF
LINE (gxxe(hj + 1), 110)-(gxxe(hj + 1) + 1, 210), , BF
LINE (gxxe(hj + 1), 150)-(gxxe(hj + 1) + 1, 210), , BF
LINE (gyye(hj + 1), 210)-(gyye(hj + 1) + 1, 240), , BF
LINE (gxxe(hj + 2), 240)-(gxxe(hj + 2) + 1, 340), , BF
LINE (gxxe(hj + 2), 280)-(gxxe(hj + 2) + 1, 340), , BF
LINE (gyye(hj + 2), 340)-(gyye(hj + 2) + 1, 370), , BF
LINE (gzze(hj + 1), 150)-(gzze(hj + 1) + 1, 280), , BF
LINE (gyy1(hj), 110)-(gyy1(hj) + 1, 150), , BF
COLOR 5
LINE (gzze(hj), 150)-(gyy1(hj), 155), , BF
LINE (gzze(hj + 1), 280)-(gzz1(hj + 1), 285), , BF
LINE (gzze(hj + 2), 410)-(gzz1(hj + 2), 415), , BF
LINE (gzze(hj + 2), 280)-(gzze(hj + 2) + 1, 410), , BF
IF gxxee(4) > 480 * (page - 1) AND gxxee(4) < 480 * (page)
THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
COLOR 14
IF nof = 1 THEN
LINE (po, 115)-(po + 2, 125), , BF
ELSEIF nof = 2 THEN
LINE (po, 245)-(po + 2, 255), , BF
ELSEIF nof = 3 THEN
LINE (po, 175)-(po + 2, 185), , BF
END IF
END IF
IF gxxeei(4) > 480 * (page - 1) AND gxxeei(4) < 480 * (page)
THEN
poi = gxxeei(4)
IF poi > gzze(4) THEN
poi = gzze(4)
END IF
DO UNTIL poi < 480
poi = poi - 480
```

```
LOOP
COLOR 9
IF nof1 = 1 THEN
LINE (poi, 155)-(poi + 2, 170), , BF
ELSEIF nof1 = 2 THEN
LINE (poi, 285)-(poi + 2, 300), , BF
ELSEIF nof1 = 3 THEN
LINE (poi, 415)-(poi + 2, 430), , BF
END IF
END IF
SUB plpl2 (gxxee(), gxxeei(), page, nof, nof1, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), hj1) STATIC

hj1 = 6
COLOR 5
LINE (qxxe(hj1), 80)-(qxx(hj1), 85), , BF
LINE (qyye(hj1), 110)-(qyy(hj1), 115), , BF
LINE (qxxe(hj1 + 1), 210)-(qxx(hj1 + 1), 215), , BF
LINE (qyye(hj1 + 1), 240)-(qyy(hj1 + 1), 245), , BF
LINE (qxxe(hj1 + 2), 340)-(qxx(hj1 + 2), 345), , BF
LINE (qyye(hj1 + 2), 370)-(qyy(hj1 + 2), 375), , BF
LINE (qzze(hj1), 150)-(qzz(hj1), 155), , BF
LINE (qzze(hj1 + 1), 280)-(qzz(hj1 + 1), 285), , BF
LINE (qzze(hj1 + 2), 410)-(qzz(hj1 + 2), 415), , BF
COLOR 4
IF qyye(hj1) = 0 THEN
LINE (0, 0)-(0, 0)
ELSE
LINE (qyye(hj1), 80)-(qyye(hj1) + 1, 110), , BF
END IF
IF qxxe(hj1 + 1) = 0 THEN
LINE (0, 0)-(0, 0)
ELSE
LINE (qxxe(hj1 + 1), 110)-(qxxe(hj1 + 1) + 1, 210), , BF
LINE (qxxe(hj1 + 1), 150)-(qxxe(hj1 + 1) + 1, 210), , BF
END IF
IF qyye(hj1 + 1) = 0 THEN
LINE (0, 0)-(0, 0)
ELSE
LINE (qyye(hj1 + 1), 210)-(qyye(hj1 + 1) + 1, 240), , BF
END IF
IF qxxe(hj1) = 0 THEN
LINE (0, 0)-(0, 0)
ELSE
LINE (qxxe(hj1 + 2), 240)-(qxxe(hj1 + 2) + 1, 340), , BF
LINE (qxxe(hj1 + 2), 280)-(qxxe(hj1 + 2) + 1, 340), , BF
END IF
IF qyye(hj1 + 2) = 0 THEN
LINE (0, 0)-(0, 0)
ELSE
LINE (qyye(hj1 + 2), 340)-(qyye(hj1 + 2) + 1, 370), , BF
END IF
IF qzze(hj1) = 0 THEN
LINE (0, 0)-(0, 0)
ELSE
LINE (qzze(hj1), 110)-(qzze(hj1) + 1, 150), , BF
END IF
IF qzze(hj1 + 1) = 0 THEN
```

```
LINE (0, 0)-(0, 0)
ELSE
LINE (qzze(hj1 + 1), 240)-(qzze(hj1 + 1) + 1, 280), , BF
END IF
IF qzze(hj1 + 2) = 0 THEN
LINE (0, 0)-(0, 0)
ELSE
LINE (qzze(hj1 + 2), 370)-(qzze(hj1 + 2) + 1, 410), , BF
END IF
COLOR 15
IF gxxee(4) > 480 * (page - 1) AND gxxee(4) < 480 * (page)
THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
COLOR 14
IF nof = 3 THEN
LINE (po, 110)-(po + 2, 120), , BF
ELSEIF nof = 4 THEN
LINE (po, 240)-(po + 2, 250), , BF
ELSEIF nof = 5 THEN
LINE (po, 370)-(po + 2, 380), , BF
END IF
END IF

IF gxxeei(4) > 480 * (page - 1) AND gxxeei(4) < 480 * (page)
THEN
poi = gxxeei(4)
IF poi > gzze(4) THEN
poi = gzze(4)
END IF
DO UNTIL poi < 480
poi = poi - 480
LOOP
COLOR 1
IF nof1 = 3 THEN
LINE (poi, 150)-(poi + 2, 160), , BF
ELSEIF nof1 = 4 THEN
LINE (poi, 280)-(poi + 2, 290), , BF
ELSEIF nof1 = 5 THEN
LINE (poi, 410)-(poi + 2, 420), , BF
END IF
END IF
COLOR 15

END SUB
```

```
SUB p1p12g (gxxe(), gxx1(), gyye(), gyy1(), gzze(), gzz1(),
i, a) STATIC

i = a + 1
COLOR 5
LINE (gxxe(i), 80)-(gxx1(i), 85), , BF
LINE (gyye(i), 110)-(gyy1(i), 115), , BF
LINE (gzze(i), 150)-(gzz1(i), 155), , BF
LINE (gxxe(i + 1), 210)-(gxx1(i + 1), 215), , BF
LINE (gyye(i + 1), 240)-(gyy1(i + 1), 245), , BF
LINE (gzze(i + 1), 280)-(gzz1(i + 1), 285), , BF
LINE (gxxe(i + 2), 340)-(gxx1(i + 2), 345), , BF
LINE (gyye(i + 2), 370)-(gyy1(i + 2), 375), , BF
LINE (gzze(i + 2), 410)-(gzz1(i + 2), 415), , BF
COLOR 4
LINE (gyye(i), 80)-(gyye(i) + 1, 110), , BF
LINE (gxxe(i + 1), 110)-(gxxe(i + 1) + 1, 210), , BF
LINE (gxxe(i + 1), 150)-(gxxe(i + 1) + 1, 210), , BF
LINE (gyye(i + 1), 210)-(gyye(i + 1) + 1, 240), , BF
LINE (gxxe(i + 2), 240)-(gxxe(i + 2) + 1, 340), , BF
LINE (gxxe(i + 2), 280)-(gxxe(i + 2) + 1, 340), , BF
LINE (gyye(i + 2), 340)-(gyye(i + 2) + 1, 370), , BF
LINE (gzze(i), 110)-(gzze(i) + 1, 150), , BF
LINE (gzze(i + 1), 150)-(gzze(i + 1) + 1, 280), , BF
LINE (gzze(i + 2), 280)-(gzze(i + 2) + 1, 410), , BF
COLOR 15

END SUB
```

```
SUB p1p2 (gxxe(), gxx1(), gyye(), gyy1(), gzze(), gzz1(),
hj1) STATIC
hj1 = 6
COLOR 5
LINE (gxxe(hj1), 80)-(gxx1(hj1), 85), , BF
LINE (gyye(hj1), 110)-(gyy1(hj1), 115), , BF
LINE (gzze(hj1), 150)-(gzz1(hj1), 155), , BF
LINE (gxxe(hj1 + 1), 210)-(gxx1(hj1 + 1), 215), , BF
LINE (gyye(hj1 + 1), 240)-(gyy1(hj1 + 1), 245), , BF
LINE (gzze(hj1 + 1), 280)-(gzz1(hj1 + 1), 285), , BF
LINE (gxxe(hj1 + 2), 340)-(gxx1(hj1 + 2), 345), , BF
LINE (gyye(hj1 + 2), 370)-(gyy1(hj1 + 2), 375), , BF
COLOR 4
LINE (gyye(hj1), 80)-(gyye(hj1) + 1, 110), , BF
LINE (gxxe(hj1 + 1), 110)-(gxxe(hj1 + 1) + 1, 210), , BF
LINE (gxxe(hj1 + 1), 150)-(gxxe(hj1 + 1) + 1, 210), , BF
LINE (gyye(hj1 + 1), 210)-(gyye(hj1 + 1) + 1, 240), , BF
LINE (gxxe(hj1 + 2), 240)-(gxxe(hj1 + 2) + 1, 340), , BF
LINE (gxxe(hj1 + 2), 280)-(gxxe(hj1 + 2) + 1, 340), , BF
LINE (gyye(hj1 + 2), 340)-(gyye(hj1 + 2) + 1, 370), , BF
LINE (gzze(hj1), 110)-(gzze(hj1) + 1, 150), , BF
LINE (gzze(hj1 + 1), 150)-(gzze(hj1 + 1) + 1, 280), , BF
COLOR 15
END SUB
```

```
SUB p1p2g (gxxe(), gxx1(), gyye(), gyy1(), gzze(), gzz1(), i,
a) STATIC

i = a + 1
COLOR 5
LINE (gxxe(i), 80)-(gxx1(i), 85), , BF
LINE (gyye(i), 110)-(gyy1(i), 115), , BF
LINE (gzze(i), 150)-(gzz1(i), 155), , BF
LINE (gxxe(i + 1), 210)-(gxx1(i + 1), 215), , BF
LINE (gyye(i + 1), 240)-(gyy1(i + 1), 245), , BF
LINE (gzze(i + 1), 280)-(gzz1(i + 1), 285), , BF
LINE (gxxe(i + 2), 340)-(gxx1(i + 2), 345), , BF
LINE (gyye(i + 2), 370)-(gyy1(i + 2), 375), , BF
COLOR 4
LINE (gyye(i), 80)-(gyye(i) + 1, 110), , BF
LINE (gxxe(i + 1), 110)-(gxxe(i + 1) + 1, 210), , BF
LINE (gxxe(i + 1), 150)-(gxxe(i + 1) + 1, 210), , BF
LINE (gyye(i + 1), 210)-(gyye(i + 1) + 1, 240), , BF
LINE (gxxe(i + 2), 240)-(gxxe(i + 2) + 1, 340), , BF
LINE (gxxe(i + 2), 280)-(gxxe(i + 2) + 1, 340), , BF
LINE (gyye(i + 2), 340)-(gyye(i + 2) + 1, 370), , BF
LINE (gzze(i), 110)-(gzze(i) + 1, 150), , BF
LINE (gzze(i + 1), 150)-(gzze(i + 1) + 1, 280), , BF
COLOR 15


END SUB
```

```
SUB p2p11 (gxxee(), gxxeei(), page, nof1, nof, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), hj2) STATIC
hj2 = 4
COLOR 5
LINE (qxxe(hj2), 80)-(qxx(hj2), 85), , BF
LINE (qyye(hj2), 110)-(qyy(hj2), 115), , BF
LINE (qzze(hj2), 150)-(qzz(hj2) + 1, 155), , BF
LINE (qxxe(hj2 + 1), 210)-(qxx(hj2 + 1), 215), , BF
LINE (qyye(hj2 + 1), 240)-(qyy(hj2 + 1), 245), , BF
LINE (qzze(hj2 + 1), 280)-(qzz(hj2 + 1) + 1, 285), , BF
LINE (qxxe(hj2 + 2), 340)-(qxx(hj2 + 2), 345), , BF
LINE (qyye(hj2 + 2), 370)-(qyy(hj2 + 2), 375), , BF
LINE (qzze(hj2 + 2), 410)-(qzz(hj2 + 2) + 1, 415), , BF
COLOR 4
IF qyye(hj2) = 0 THEN
LINE (qyye(hj2), 80)-(qyye(hj2), 110), , BF
ELSE
LINE (qyye(hj2), 80)-(qyye(hj2) + 1, 110), , BF
END IF
IF qxxe(hj2 + 1) = 0 THEN
LINE (qxxe(hj2 + 1), 80)-(qxxe(hj2 + 1), 110), , BF
ELSE
LINE (qxxe(hj2 + 1), 110)-(qxxe(hj2 + 1) + 1, 210), , BF
LINE (qxxe(hj2 + 1), 150)-(qxxe(hj2 + 1) + 1, 210), , BF
END IF
IF qyye(hj2 + 1) = 0 THEN
LINE (qyye(hj2 + 1), 80)-(qyye(hj2 + 1), 110), , BF
ELSE
LINE (qyye(hj2 + 1), 210)-(qyye(hj2 + 1) + 1, 240), , BF
END IF
IF qxxe(hj2 + 2) = 0 THEN
LINE (qxxe(hj2 + 2), 80)-(qxxe(hj2 + 2), 110), , BF
ELSE
LINE (qxxe(hj2 + 2), 240)-(qxxe(hj2 + 2) + 1, 340), , BF
LINE (qxxe(hj2 + 2), 280)-(qxxe(hj2 + 2) + 1, 340), , BF
END IF
IF qyye(hj2 + 2) = 0 THEN
LINE (qye(hj2 + 2), 80)-(qyye(hj2 + 2), 110), , BF
ELSE
LINE (qyye(hj2 + 2), 340)-(qyye(hj2 + 2) + 1, 370), , BF
END IF
COLOR 4
IF qzze(hj2) = 0 THEN
LINE (0, 0)-(0, 0), , BF
```

```
ELSE
LINE (qzze(hj2), 110)-(qzze(hj2) + 1, 150), , BF
END IF
IF qzze(hj2 + 1) = 0 THEN
LINE (0, 0)-(0, 0), , BF
ELSE
LINE (qzze(hj2 + 1), 240)-(qzze(hj2 + 1) + 1, 280), , BF
END IF
IF qzze(hj2 + 2) = 0 THEN
LINE (0, 0)-(0, 0), , BF
ELSE
LINE (qzze(hj2 + 2), 370)-(qzze(hj2 + 2) + 1, 410), , BF
END IF
IF gxxee(4) > 480 * (page - 1) AND gxxee(4) < 480 * (page)
THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
COLOR 14
IF nof = 1 THEN
LINE (po, 110)-(po + 2, 120), , BF
ELSEIF nof = 2 THEN
LINE (po, 240)-(po + 2, 250), , BF
ELSEIF nof = 3 THEN
LINE (po, 370)-(po + 2, 380), , BF
END IF
END IF
IF gxxeei(4) > 480 * (page - 1) AND gxxeei(4) < 480 * (page)
THEN
poi = gxxeei(4)
IF poi > gzze(4) THEN
poi = gzze(4)
END IF
DO UNTIL poi < 480
poi = poi - 480
LOOP
COLOR 1
IF nof1 = 1 THEN
LINE (poi, 150)-(poi + 2, 160), , BF
ELSEIF nof1 = 2 THEN
LINE (poi, 280)-(poi + 2, 290), , BF
ELSEIF nof1 = 3 THEN
LINE (poi, 410)-(poi + 2, 420), , BF
END IF
END IF
COLOR 15
END SUB
```

```
SUB p2p12 (qxxe(), qxx(), qyye(), qyy(), qzze(), qzz(), i, a)
STATIC

i = a + 1
COLOR 5
LINE (qxxe(i), 80)-(qxx(i), 85), , BF
LINE (qyye(i), 110)-(qyy(i), 115), , BF
LINE (qzze(i), 150)-(qzz(i) + 1, 155), , BF
LINE (qxxe(i + 1), 210)-(qxx(i + 1), 215), , BF
LINE (qyye(i + 1), 240)-(qyy(i + 1), 245), , BF
LINE (qzze(i + 1), 280)-(qzz(i + 1) + 1, 285), , BF
LINE (qxxe(i + 2), 340)-(qxx(i + 2), 345), , BF
LINE (qyye(i + 2), 370)-(qyy(i + 2), 375), , BF
LINE (qzze(i + 2), 410)-(qzz(i + 2) + 1, 415), , BF
COLOR 4
LINE (qyye(i), 80)-(qyye(i) + 1, 110), , BF
LINE (qxxe(i + 1), 110)-(qxxe(i + 1) + 1, 210), , BF
LINE (qxxe(i + 1), 150)-(qxxe(i + 1) + 1, 210), , BF
LINE (qyye(i + 1), 210)-(qyye(i + 1) + 1, 240), , BF
LINE (qxxe(i + 2), 240)-(qxxe(i + 2) + 1, 340), , BF
LINE (qxxe(i + 2), 280)-(qxxe(i + 2) + 1, 340), , BF
LINE (qyye(i + 2), 340)-(qyye(i + 2) + 1, 370), , BF

IF qzze(i) = 0 THEN
LINE (qzze(i), 80)-(qzze(i), 110), , BF
ELSE
LINE (qzze(i), 110)-(qzze(i) + 1, 150), , BF
END IF
IF qzze(i + 1) = 0 THEN
LINE (qzze(i + 1), 80)-(qzze(i + 1), 110), , BF
ELSE
LINE (qzze(i + 1), 240)-(qzze(i + 1) + 1, 280), , BF
END IF
IF qzze(i + 2) = 0 THEN
LINE (qzze(i + 2), 80)-(qzze(i + 2), 110), , BF
ELSE
LINE (qzze(i + 2), 370)-(qzze(i + 2) + 1, 410), , BF
END IF

COLOR 15

eND SUB
```

```
SUB p2p22 (gxxee(), gxxeei(), nof, nof1, page, qxxe(), qxx(),
qyye(), qyy(), qzze(), qzz(), i, a) STATIC
i = a + 1
COLOR 5
LINE (qxxe(i), 80)-(qxx(i), 85), , BF
LINE (qyye(i), 110)-(qyy(i), 115), , BF
LINE (qzze(i), 150)-(qzz(i) + 1, 155), , BF
LINE (qxxe(i + 1), 210)-(qxx(i + 1), 215), , BF
LINE (qyye(i + 1), 240)-(qyy(i + 1), 245), , BF
LINE (qzze(i + 1), 280)-(qzz(i + 1) + 1, 285), , BF
LINE (qxxe(i + 2), 340)-(qxx(i + 2), 345), , BF
LINE (qyye(i + 2), 370)-(qyy(i + 2), 375), , BF
COLOR 4
LINE (qyye(i), 80)-(qyye(i) + 1, 110), , BF
LINE (qxxe(i + 1), 110)-(qxxe(i + 1) + 1, 210), , BF
LINE (qxxe(i + 1), 150)-(qxxe(i + 1) + 1, 210), , BF
LINE (qyye(i + 1), 210)-(qyye(i + 1) + 1, 240), , BF
LINE (qxxe(i + 2), 240)-(qxxe(i + 2) + 1, 340), , BF
LINE (qxxe(i + 2), 280)-(qxxe(i + 2) + 1, 340), , BF
LINE (qyye(i + 2), 340)-(qyye(i + 2) + 1, 370), , BF
LINE (qzze(i), 110)-(qzze(i) + 1, 150), , BF
LINE (qzze(i + 1), 240)-(qzze(i + 1) + 1, 280), , BF

IF gxxee(4) > 480 * (page - 1) AND gxxee(4) < 480 * (page)
THEN
po = gxxee(4)
DO UNTIL po < 480
po = po - 480
LOOP
COLOR 14
IF nof = 3 THEN
LINE (po, 110)-(po + 2, 120), , BF
ELSEIF nof = 4 THEN
LINE (po, 240)-(po + 2, 250), , BF
ELSEIF nof = 5 THEN
LINE (po, 370)-(po + 2, 380), , BF
END IF
END IF
IF gxxeei(4) > 480 * (page - 1) AND gxxeei(4) < 480 * (page)
THEN
poi = gxxeei(4)
```

```
IF poi > gzze(4) THEN
poi = gzze(4)
END IF
DO UNTIL poi < 480
poi = poi - 480
LOOP
COLOR 1
IF nof1 = 3 THEN
LINE (poi, 150)-(poi + 2, 160), , BF
ELSEIF nof1 = 4 THEN
LINE (poi, 280)-(poi + 2, 290), , BF
ELSEIF nof1 = 5 THEN
LINE (poi, 410)-(poi + 2, 420), , BF
END IF
END IF
COLOR 15
END SUB

SUB sf (afin1(), afin2(), afin3(), afin4(), afin5(), afin6(),
afin7(), afin8(), afin9(), afin10(), aser1(), aser2(),
aser3(), aser4(), aser5(), aser6(), i) STATIC

        COLOR 5
        LINE (afin1(i + 1), 290)-(afin2(i + 1), 295), , BF
        LINE (afin3(i + 1), 308)-(afin4(i + 1), 313), , BF
        LINE (afin5(i + 1), 326)-(afin6(i + 1), 331), , BF
        LINE (afin7(i + 1), 344)-(afin8(i + 1), 349), , BF
        LINE (afin9(i + 1), 362)-(afin10(i + 1), 367), , BF
        LINE (aser1(i + 1), 390)-(aser2(i + 1), 395), , BF
        LINE (aser3(i + 1), 408)-(aser4(i + 1), 413), , BF
        LINE (aser5(i + 1), 424)-(aser6(i + 1), 429), , BF

END SUB
```

APPENDIX 15 CODING FOR THE OPTIMISATION MODEL

```
DECLARE SUB ratio1 (s1!, s2!, s162!, s188!, s465!, f1!, f2!,
f3!, f4!, costra!, ib219!, ib217!, b219!, b217!, s1m!, s2m!,
s162m!, s188m!, s465m!, f1m!, f2m!, f3m!, f4m!, c1!, ib219m!,
ib217m!, b219m!, b217m!, DIFF!(), ratio!(), indact!(),
indactm!(), dir!())
DECLARE SUB ratio2 (f6!, f7!, f8!, f9!, f10!, s4!, sem1!,
see1!, tf374!, tf377!, tf380!, tf373!, tf376!, tf379!,
tf375!, tf378!, tf381!, f6m!, f7m!, f8m!, f9m!, f10m!, s4m!,
sem1m!, see1m!, tf374m!, tf377m!, tf380m!, tf373m!, tf376m!,
tf379m!, tf375m!, TF378M!, tf381m!, DIFF!(), ratio!(),
indact!(), indactm!(), dir!(), nofloor)
DECLARE SUB xyy (dur!(), lag!(), X!(), Y!(), nofloor!, NOF!,
NOF1!, PR!)
DECLARE SUB XYYM (DURM!(), lagm!(), XM!(), YM!(), nofloor!,
NOFm!, NOF1m!, prM!)
DECLARE SUB lagii (l1lr!, lrlb!, lblt!, l1lc!, f1f2!, f2f3!,
lagsifo!, lagfogr!, laggrfir!, lagcwsb1!, lagsbs1!, lagcw2!,
lagcw1!, lagsb1!, lagsb2!, lagsc1!, lagsc2!, laglaro!,
lagnfbb!, lagi!, lrlb1!, lagra!, lain!, lawa!, lafl!, lace!,
laex!, lali!, lame!, lael!, lag!())
DECLARE SUB durn (s5!, s6!, s163!, s188!, s465!, F11!, F12!,
f13!, F14!, dayra!, ib169!, ib161!, b169!, b161!, fii2!,
fiw2!, fif2!, fic2!, fie2!, sel2!, sem2!, see2!, tf243!,
tf286!, tf303!, tf28!, tf71!, tf111!, tf149!, TF176!, tf150!,
dur!())
DECLARE SUB lagiim (l1lrm!, lrlbm!, lbltm!, l1lcm!, f1f2m!,
f2f3m!, lagsifom!, lagfogrm!, laggrfirm!, lagcwsb1m!,
lagsbs1m!, lagcw2m!, lagcw1m!, lagsb1m!, lagsb2m!, lagsc1m!,
lagsc2m!, laglarom!, lagnfbbm!, lagim!, lrlb1m!, lagram!,
lainm!, lawam!, laflm!, lacem!, laexm!, lalim!, lamem!,
laelm!, lagm!())
DECLARE SUB du2 (s5m!, s6m!, s163m!, s188m!, s465m!, F11m!,
F12m!, f13m!, F14m!, dayram!, ib169m!, ib161m!, b169m!,
b161m!, fii2m!, fiw2m!, fif2m!, fic2m!, fie2m!, sel2m!,
sem2m!, see2m!, tf243m!, tf286m!, tf303m!, tf28m!, tf71m!,
tf111m!, tf149m!, TF176M!, tf150m!, DURM!())
DECLARE SUB diff1 (s5!, s6!, s163!, s188!, s465!, F11!, F12!,
f13!, F14!, dayra!, ib169!, ib161!, b169!, b161!, s5m!, s6m!,
s163m!, s188m!, s465m!, F11m!, F12m!, f13m!, F14m!, dayram!,
ib169m!, ib161m!, b169m!, b161m!, DIFF!())
DECLARE SUB diff2 (fii2!, fiw2!, fif2!, fic2!, fie2!, sel2!,
sem2!, see2!, tf243!, tf286!, tf303!, tf28!, tf71!, tf111!,
tf149!, TF176!, tf150!, fii2m!, fiw2m!, fif2m!, fic2m!,
fie2m!, sel2m!, sem2m!, see2m!, tf243m!, tf286m!, tf303m!,
tf28m!, tf71m!, tf111m!, tf149m!, TF176M!, tf150m!, DIFF!())

OPEN "tott.dat" FOR INPUT AS #10
INPUT #10, PR, prM, projdir, projdirm, INDIRECT, INDIRECTM,
totco, totcom, dur1, dur2, dur3, dur1m, dur2m, dur3m, d31,
d32, d33, d34, d35, d36, d37, d38, d31m, d32m, d33m, d34m,
d35m, d36m, d37m, d38m


DIM lag(29), lagm(29), DIFF(31), ratio(150), dur(31),
DURM(31), X(150), Y(150), indact(150), XM(150), YM(150),
indactm(150), dir(31)
```

```
OPEN "nooo.dat" FOR INPUT AS #1
INPUT #1, nofloor
CLOSE 1
OPEN "lagmaind.dat" FOR INPUT AS #8
INPUT #8, ltlr, lrlb, lblt, ltlc, f1f2, f2f3, lagsifo,
lagfogr, laggrfir, lagcwsb1, lagsbs1, lagcw2, lagcw1, lagsb1,
lagsb2, lagsc1, lagsc2, laglaro, lagnfbb, lagi, NOF, NOF1,
lrlb1, lagra, lain, lawa, lafl, lace, laex, lali, lame, lael

OPEN "lagdm.dat" FOR INPUT AS #18
INPUT #18, ltlrm, lrlbm, lbltm, ltlcm, f1f2m, f2f3m,
lagsifom, lagfogrm, laggrfirm, lagcwsblm, lagsbslm, lagcw2m,
lagcwlm, lagsblm, lagsb2m, lagsclm, lagsc2m, laglarom,
lagnfbbm, lagim, NOFm, NOF1m, lrlblm, lagram, lainm, lawam,
laflm, lacem, laexm, lalim, lamem, laelm

OPEN "coptsi.dat" FOR INPUT AS #9
INPUT #9, sitotc, sdat, sitotcm, sdatm, s5m, s6m, s163m,
s188m, s465m, slm, s2m, s162m, s187m, s464m, s1, s5, s8m,
s20m, s22m, s24m, s82m, s133m, s158m, s183m, s462m, s6, s163,
s188, s465, s2, s162, s187, s464, s8, s20, s22, s24, s82,
s133, s158, s183, s462, comp1, comp1m
CLOSE 9

OPEN "coptsi2.dat" FOR OUTPUT AS #9
PRINT #9, USING "#######.#"; sitotc; sdat; sitotcm; sdatm;
s5m; s6m; s163m; s188m; s465m; slm; s2m; s162m; s187m; s464m;
s1; s5; s8m; s20m; s22m; s24m; s82m; s133m; s158m; s183m;
s462m; s6; s163; s188; s465; s2; s162; s187; s464; s8; s20;
s22; s24; s82; s133; s158; s183; s462; comp1; comp1m
CLOSE 9

OPEN "coptsi2.dat" FOR INPUT AS #9
INPUT #9, sitotc, sdat, sitotcm, sdatm, s5m, s6m, s163m,
s188m, s465m, slm, s2m, s162m, s187m, s464m, s1, s5, s8m,
s20m, s22m, s24m, s82m, s133m, s158m, s183m, s462m, s6, s163,
s188, s465, s2, s162, s187, s464, s8, s20, s22, s24, s82,
s133, s158, s183, s462, comp1, comp1m
CLOSE 9

OPEN "coptfn.dat" FOR INPUT AS #20
INPUT #20, cofotot, fodat, cofototm, fodatm, f24m, F12m,
f13m, F14m, f2, f3, f4, F14, f2m, f3m, f180m, f125m, fgam,
f14b, f14bm, f1, f1m, f15, F11, F11m, f461b, F12, f13
CLOSE 20

OPEN "coptfn2.dat" FOR OUTPUT AS #20
PRINT #20, USING "#######.#"; cofotot; fodat; cofototm;
fodatm; f24m; F12m; f13m; F14m; f2; f3; f4; F14; f2m; f3m;
f180m; f125m; fgam; f14b; f14bm; f1; f1m; f15; F11; F11m;
f461b; F12; f13
CLOSE 20


OPEN "coptfn2.dat" FOR INPUT AS #20
INPUT #20, cofotot, fodat, cofototm, fodatm, f24m, F12m,
f13m, F14m, f2, f3, f4, F14, f2m, f3m, f180m, f125m, fgam,
f14b, f14bm, f1, f1m, f15, F11, F11m, f461b, F12, f13
CLOSE 20
```

```
OPEN "coptf1.dat" FOR INPUT AS #11
INPUT #11, fltcos, fdat, tf373, tf374, tf375, tf376, tf377,
tf378, tf379, tf380, tf381, tf28, tf149, tf243, tf71, TF176,
tf286, tf111, tf150, tf303, tf373m, tf374m, tf375m, tf376m,
tf377m, TF378M, tf379m, tf380m, tf381m, tf28m, tf149m,
tf243m, tf71m, TF176M, tf286m, tf111m, tf150m, tf303m
CLOSE 11

OPEN "coptf12.dat" FOR OUTPUT AS #11
PRINT #11, USING "#######.#"; fltcos; fdat; tf373; tf374;
tf375; tf376; tf377; tf378; tf379; tf380; tf381; tf28; tf149;
tf243; tf71; TF176; tf286; tf111; tf150; tf303; tf373m;
tf374m; tf375m; tf376m; tf377m; TF378M; tf379m; tf380m;
tf381m; tf28m; tf149m; tf243m; tf71m; TF176M; tf286m; tf111m;
tf150m; tf303m
CLOSE 11

OPEN "coptf12.dat" FOR INPUT AS #11
INPUT #11, fltcos, fdat, tf373, tf374, tf375, tf376, tf377,
tf378, tf379, tf380, tf381, tf28, tf149, tf243, tf71, TF176,
tf286, tf111, tf150, tf303, tf373m, tf374m, tf375m, tf376m,
tf377m, TF378M, tf379m, tf380m, tf381m, tf28m, tf149m,
tf243m, tf71m, TF176M, tf286m, tf111m, tf150m, tf303m
CLOSE 11

OPEN "coptbb.dat" FOR INPUT AS #12
INPUT #12, ib2, b172, ib2m, b173m, b159m, b167m, b217, b219,
b161, b169, b217m, b219m, b161m, b169m
CLOSE 12

OPEN "coptbb2.dat" FOR OUTPUT AS #12
PRINT #12, USING "#######.#"; ib2; b172; ib2m; b173m; b159m;
b167m; b217; b219; b161; b169; b217m; b219m; b161m; b169m
CLOSE 12

OPEN "coptbb2.dat" FOR INPUT AS #12
INPUT #12, ib2, b172, ib2m, b173m, b159m, b167m, b217, b219,
b161, b169, b217m, b219m, b161m, b169m
CLOSE 12

OPEN "icoptbb.dat" FOR INPUT AS #13
INPUT #13, ib1, ib172, ib1m, ib173m, ib159m, ib167m, ib217,
ib219, ib161, ib169, ib217m, ib219m, ib161m, ib169m
CLOSE 13

OPEN "icoptbb2.dat" FOR OUTPUT AS #13
PRINT #13, USING "#######.#"; ib1; ib172; ib1m; ib173m;
ib159m; ib167m; ib217; ib219; ib161; ib169; ib217m; ib219m;
ib161m; ib169m
CLOSE 13

OPEN "icoptbb2.dat" FOR INPUT AS #13
INPUT #13, ib1, ib172, ib1m, ib173m, ib159m, ib167m, ib217,
ib219, ib161, ib169, ib217m, ib219m, ib161m, ib169m
CLOSE 13

OPEN "ra.dat" FOR INPUT AS #14
INPUT #14, costra, dayra, c1, dayram, weekra, durra
```

```
CLOSE 14

OPEN "ra2.dat" FOR OUTPUT AS #14
PRINT #14, USING "#######.#"; costra; dayra; c1; dayram;
weekra; durra
CLOSE 14

OPEN "ra2.dat" FOR INPUT AS #14
INPUT #14, costra, dayra, c1, dayram, weekra, durra
CLOSE 14

OPEN "graphfi.dat" FOR INPUT AS #15
INPUT #15, fii2, fiw2, fif2, fic2, fie2, f6, f7, f8, f9, f10,
fii2m, fiw2m, fif2m, fic2m, fie2m, f6m, f7m, f8m, f9m, f10m
CLOSE 15

OPEN "graphfi2.dat" FOR OUTPUT AS #15
PRINT #15, USING "#######.#"; fii2; fiw2; fif2; fic2; fie2;
f6; f7; f8; f9; f10; fii2m; fiw2m; fif2m; fic2m; fie2m; f6m;
f7m; f8m; f9m; f10m
CLOSE 15

OPEN "graphfi2.dat" FOR INPUT AS #15
INPUT #15, fii2, fiw2, fif2, fic2, fie2, f6, f7, f8, f9, f10,
fii2m, fiw2m, fif2m, fic2m, fie2m, f6m, f7m, f8m, f9m, f10m
CLOSE 15

OPEN "graphs.dat" FOR INPUT AS #16
INPUT #16, sel2, sem2, see2, s4, sem1, see1, sel2m, sem2m,
see2m, s4m, sem1m, see1m
CLOSE 16

OPEN "graphs2.dat" FOR OUTPUT AS #16
PRINT #16, USING "#######.#"; sel2; sem2; see2; s4; sem1;
see1; sel2m; sem2m; see2m; s4m; sem1m; see1m
CLOSE 16

OPEN "graphs2.dat" FOR INPUT AS #16
INPUT #16, sel2, sem2, see2, s4, sem1, see1, sel2m, sem2m,
see2m, s4m, sem1m, see1m
CLOSE 16

IF s163 > s188 AND s163 > s465 THEN
S7 = s163
ELSEIF s188 > s163 AND s188 > s465 THEN
S7 = s188
ELSEIF s465 > s163 AND s465 > s188 THEN
S7 = s465
END IF


f17 = tf150 + lagsc1 + lagsc2
f18 = tf111 + lagcw2 + lagcw1
f19 = tf303 + lagsb1 + lagsb2

s3 = s162 + s187 + s464

tf2 = tf373 + tf376 + tf379
tf1 = tf374 + tf377 + tf380
```

```
tf5 = tf375 + tf378 + tf381
IF s163m > s188m AND s163m > s465m THEN
S7m = s163m
ELSEIF s188m > s163m AND s188m > s465m THEN
S7m = s188m
ELSEIF s465m > s163m AND s465m > s188m THEN
S7m = s465m
END IF

f17m = tf150m + lagsc1 + lagsc2
f18m = tf111m + lagcw2 + lagcw1
f19m = tf303m + lagsb1 + lagsb2

s3m = s162m + s187m + s464m

tf2m = tf373m + tf376m + tf379m
tf1m = tf374m + tf377m + tf380m

tf5m = tf375m + TF378M + tf381m

cvb = (nofloor - 2) * 2

CALL durn(s5, s6, s163, s188, s465, F11, F12, f13, F14,
dayra, ib169, ib161, b169, b161, fii2, fiw2, fif2, fic2,
fie2, sel2, sem2, see2, tf243, tf286, tf303, tf28, tf71,
tf111, tf149, TF176, tf150, dur())

CALL du2(s5m, s6m, s163m, s188m, s465m, F11m, F12m, f13m,
F14m, dayram, ib169m, ib161m, b169m, b161m, fii2m, fiw2m,
fif2m, fic2m, fie2m, sel2m, sem2m, see2m, tf243m, tf286m,
tf303m, tf28m, tf71m, tf111m, tf149m, TF176M, tf150m, DURM())

CALL lagii(ltlr, lrlb, lblt, ltlc, f1f2, f2f3, lagsifo,
lagfogr, laggrfir, lagcwsb1, lagsbs1, lagcw2, lagcw1, lagsb1,
lagsb2, lagsc1, lagsc2, laglaro, lagnfbb, lagi, lrlb1, lagra,
lain, lawa, lafl, lace, laex, lali, lame, lael, lag())

CALL lagiim(ltlrm, lrlbm, lbltm, ltlcm, f1f2m, f2f3m,
lagsifom, lagfogrm, laggrfirm, lagcwsb1m, lagsbs1m, lagcw2m,
lagcw1m, lagsb1m, lagsb2m, lagsc1m, lagsc2m, laglarom,
lagnfbbm, lagim, lrlb1m, lagram, lainm, lawam, laflm, lacem,
laexm, lalim, lamem, laelm, lagm())

CALL diff1(s5, s6, s163, s188, s465, F11, F12, f13, F14,
dayra, ib169, ib161, b169, b161, s5m, s6m, s163m, s188m,
s465m, F11m, F12m, f13m, F14m, dayram, ib169m, ib161m, b169m,
b161m, DIFF())

CALL diff2(fii2, fiw2, fif2, fic2, fie2, sel2, sem2, see2,
tf243, tf286, tf303, tf28, tf71, tf111, tf149, TF176, tf150,
fii2m, fiw2m, fif2m, fic2m, fie2m, sel2m, sem2m, see2m,
tf243m, tf286m, tf303m, tf28m, tf71m, tf111m, tf149m, TF176M,
tf150m, DIFF())

CALL xyy(dur(), lag(), X(), Y(), nofloor, NOF, NOF1, PR)

CALL XYYM(DURM(), lagm(), XM(), YM(), nofloor, NOFm, NOF1m,
prM)
```

```
CALL ratio1(s1, s2, s162, s187, s464, f1, f2, f3, f4, costra,
ib219, ib217, b219, b217, s1m, s2m, s162m, s187m, s464m, f1m,
f2m, f3m, f24m, c1, ib219m, ib217m, b219m, b217m, DIFF(),
ratio(), indact(), indactm(), dir())

CALL ratio2(f6, f7, f8, f9, f10, s4, sem1, see1, tf374,
tf377, tf380, tf373, tf376, tf379, tf375, tf378, tf381, f6m,
f7m, f8m, f9m, f10m, s4m, sem1m, see1m, tf374m, tf377m,
tf380m, tf373m, tf376m, tf379m, tf375m, TF378M, tf381m,
DIFF(), ratio(), indactm(), indact(), dir(), nofloor)

OPEN "opt4.dat" FOR OUTPUT AS #21
PRINT #21, PR, prM
CLOSE 21

OPEN "opt4a.dat" FOR OUTPUT AS #21
PRINT #21, PR
CLOSE 21

OPEN "EMEL.dat" FOR OUTPUT AS #7
FOR I = 1 TO 9 * nofloor + 20
PRINT #7, X(I), Y(I), XM(I), YM(I), ratio(I)
NEXT
CLOSE (7)

OPEN "opt2.dat" FOR OUTPUT AS #19
'first (x)'s'
FOR I = 1 TO 32
PRINT #19, USING "####.##"; 0
NEXT
FOR I = 1 TO 6 * nofloor - 5
PRINT #19, USING "####.##"; 0
NEXT

FOR I = 1 TO 10
PRINT #19, USING "####.##"; ratio(I)
NEXT
FOR I = 11 TO 14
PRINT #19, USING "####.##"; ratio(I) / 5
NEXT
FOR I = 15 TO 26
PRINT #19, USING "####.##"; ratio(I)
NEXT
PRINT #19, USING "####.##"; ratio(27)
PRINT #19, USING "####.##"; ratio(27)
FOR I = 29 TO 31
PRINT #19, USING "####.##"; ratio(I)
NEXT

FOR I = 1 TO nofloor - 1
FOR ii = 1 TO 3
PRINT #19, USING "####.##"; ratio(22 + ii)
NEXT
PRINT #19, USING "####.##"; ratio(26)
PRINT #19, USING "####.##"; ratio(27)
PRINT #19, USING "####.##"; ratio(27)
NEXT
PRINT #19, USING "####.##"; ratio(23)
FOR I = 1 TO 3 * (nofloor - 2)
```

```
PRINT #19, USING "####.##"; 0
NEXT

FOR I = 1 TO nofloor - 2
PRINT #19, USING "####.##"; ratio(29)
PRINT #19, USING "####.##"; ratio(30)
PRINT #19, USING "####.##"; ratio(31)
NEXT
PRINT #19, USING "####.##"; 0
CLOSE 19

OPEN "dir.dat" FOR OUTPUT AS #17
FOR I = 1 TO 31
PRINT #17, dir(I)
NEXT
PRINT #17, 0
CLOSE 17

OPEN "durn.dat" FOR OUTPUT AS #17
FOR I = 1 TO 31
PRINT #17, dur(I)
NEXT
PRINT #17, 0
CLOSE 17

OPEN "durm.dat" FOR OUTPUT AS #17
FOR I = 1 TO 31
PRINT #17, DURM(I)
NEXT
CLOSE 17

OPEN "lagaa.dat" FOR OUTPUT AS #17
FOR I = 1 TO 29
PRINT #17, lagm(I),
NEXT
CLOSE #17

OPEN "nof.dat" FOR OUTPUT AS #17
PRINT #17, NOF, NOF1, NOFm, NOF1m
CLOSE #17

OPEN "tott2.dat" FOR OUTPUT AS #7
PRINT #7, PR
PRINT #7, prM
PRINT #7, projdir
PRINT #7, projdirm
PRINT #7, INDIRECT
PRINT #7, INDIRECTM
PRINT #7, totco
PRINT #7, totcom
CLOSE 7, 17, 8, 10, 18
CHAIN "c:\opt\opt2"
END
```

```
SUB diff1 (s5, s6, s163, s188, s465, F11, F12, f13, F14,
dayra, ib169, ib161, b169, b161, s5m, s6m, s163m, s188m,
s465m, F11m, F12m, f13m, F14m, dayram, ib169m, ib161m, b169m,
b161m, DIFF()) STATIC

DIFF(1) = s5 - s5m
DIFF(2) = s6 - s6m
DIFF(3) = s163 - s163m
DIFF(4) = s188 - s188m
DIFF(5) = s465 - s465m
DIFF(6) = F11 - F11m
DIFF(7) = F12 - F12m
DIFF(8) = f13 - f13m
DIFF(9) = F14 - F14m
DIFF(10) = dayra - dayram
DIFF(11) = ib169 - ib169m
DIFF(12) = ib161 - ib161m
DIFF(13) = b169 - b169m
DIFF(14) = b161 - b161m

FOR I = 1 TO 14
IF DIFF(I) <> CINT(DIFF(I)) THEN
DIFF(I) = CINT(DIFF(I))
END IF
NEXT

END SUB
```

```
SUB diff2 (fii2, fiw2, fif2, fic2, fie2, sel2, sem2, see2,
tf243, tf286, tf303, tf28, tf71, tf111, tf149, TF176, tf150,
fii2m, fiw2m, fif2m, fic2m, fie2m, sel2m, sem2m, see2m,
tf243m, tf286m, tf303m, tf28m, tf71m, tf111m, tf149m, TF176M,
tf150m, DIFF()) STATIC

DIFF(15) = fii2 - fii2m
DIFF(16) = fiw2 - fiw2m
DIFF(17) = fif2 - fif2m
DIFF(18) = fic2 - fic2m
DIFF(19) = fie2 - fie2m
DIFF(20) = sel2 - sel2m
DIFF(21) = sem2 - sem2m
DIFF(22) = see2 - see2m
DIFF(23) = tf243 - tf243m
DIFF(24) = tf286 - tf286m
DIFF(25) = tf303 - tf303m
DIFF(26) = tf28 - tf28m
DIFF(27) = tf71 - tf71m
DIFF(28) = tf111 - tf111m
DIFF(29) = tf149 - tf149m
DIFF(30) = TF176 - TF176M
DIFF(31) = tf150 - tf150m

FOR I = 15 TO 31
IF DIFF(I) <> CINT(DIFF(I)) THEN
DIFF(I) = CINT(DIFF(I))
END IF
NEXT
END SUB
```

```
SUB du2 (s5m, s6m, s163m, s188m, s465m, F11m, F12m, f13m,
F14m, dayram, ib169m, ib161m, b169m, b161m, fii2m, fiw2m,
fif2m, fic2m, fie2m, sel2m, sem2m, see2m, tf243m, tf286m,
tf303m, tf28m, tf71m, tf111m, tf149m, TF176M, tf150m, DURM())

DURM(1) = s5m
DURM(2) = s6m
DURM(3) = s163m
DURM(4) = s188m
DURM(5) = s465m
DURM(6) = F11m
DURM(7) = F12m
DURM(8) = f13m
DURM(9) = F14m
DURM(10) = dayram
DURM(11) = ib169m
DURM(12) = ib161m
DURM(13) = b169m
DURM(14) = b161m
DURM(15) = fii2m
DURM(16) = fiw2m
DURM(17) = fif2m
DURM(18) = fic2m
DURM(19) = fie2m
DURM(20) = sel2m
DURM(21) = sem2m
DURM(22) = see2m
DURM(23) = tf243m
DURM(24) = tf286m
DURM(25) = tf303m
DURM(26) = tf28m
DURM(27) = tf71m
DURM(28) = tf111m
DURM(29) = tf149m
DURM(30) = TF176M
DURM(31) = tf150m

FOR I = 1 TO 31
IF DURM(I) - INT(DURM(I)) <> 0 THEN
DURM(I) = INT(DURM(I)) + 1
END IF
NEXT

END SUB
```

```
SUB durn (s5, s6, s163, s188, s465, F11, F12, f13, F14,
dayra, ib169, ib161, b169, b161, fii2, fiw2, fif2, fic2,
fie2, sel2, sem2, see2, tf243, tf286, tf303, tf28, tf71,
tf111, tf149, TF176, tf150, dur())

dur(1) = s5
dur(2) = s6
dur(3) = s163
dur(4) = s188
dur(5) = s465
dur(6) = F11
dur(7) = F12
dur(8) = f13
dur(9) = F14
dur(10) = dayra
dur(11) = ib169
dur(12) = ib161
dur(13) = b169
dur(14) = b161
dur(15) = fii2
dur(16) = fiw2
dur(17) = fif2
dur(18) = fic2
dur(19) = fie2
dur(20) = sel2
dur(21) = sem2
dur(22) = see2
dur(23) = tf243
dur(24) = tf286
dur(25) = tf303
dur(26) = tf28
dur(27) = tf71
dur(28) = tf111
dur(29) = tf149
dur(30) = TF176
dur(31) = tf150

FOR I = 1 TO 31
IF dur(I) - INT(dur(I)) <> 0 THEN
dur(I) = INT(dur(I)) + 1
END IF
NEXT

END SUB
```

```
SUB lagii (ltlr, lrlb, lblt, ltlc, f1f2, f2f3, lagsifo,
lagfogr, laggrfir, lagcwsb1, lagsbs1, lagcw2, lagcw1, lagsb1,
lagsb2, lagsc1, lagsc2, laglaro, lagnfbb, lagi, lrlb1, lagra,
lain, lawa, lafl, lace, laex, lali, lame, lael, lag()) STATIC

lag(1) = ltlr
lag(2) = lrlb
lag(3) = lrlb
lag(4) = lrlb
lag(5) = lrlb1
lag(6) = lagsifo
lag(7) = f1f2
lag(8) = f2f3
lag(9) = lagfogr
lag(10) = laggrfir
lag(11) = lagcwsb1
lag(12) = lagsbs1
lag(13) = lagi
lag(14) = lagnfbb
lag(15) = lagcw2
lag(16) = lagcw1
lag(17) = lagsb1
lag(18) = lagsb2
lag(19) = lagsc1
lag(20) = lagsc2
lag(21) = lain
lag(22) = lawa
lag(23) = lafl
lag(24) = lace
lag(25) = laex
lag(26) = lali
lag(27) = lame
lag(28) = lael
lag(29) = lagra

FOR I = 1 TO 29
IF lag(I) - INT(lag(I)) <> 0 THEN
lag(I) = INT(lag(I)) + 1
END IF
NEXT

END SUB
```

```
SUB lagiim (ltlrm, lrlbm, lbltm, ltlcm, f1f2m, f2f3m,
lagsifom, lagfogrm, laggrfirm, lagcwsblm, lagsbslm, lagcw2m,
lagcwlm, lagsblm, lagsb2m, lagsclm, lagsc2m, laglarom,
lagnfbbm, lagim, lrlblm, lagram, lainm, lawam, laflm, lacem,
laexm, lalim, lamem, laelm, lagm()) STATIC
lagm(1) = ltlrm
lagm(2) = lrlbm
lagm(3) = lrlbm
lagm(4) = lrlbm
lagm(5) = lrlblm
lagm(6) = lagsifom
lagm(7) = f1f2m
lagm(8) = f2f3m
lagm(9) = lagfogrm
lagm(10) = laggrfirm
lagm(11) = lagcwsblm
lagm(12) = lagsbslm
lagm(13) = lagim
lagm(14) = lagnfbbm
lagm(15) = lagcw2m
lagm(16) = lagcwlm
lagm(17) = lagsblm
lagm(18) = lagsb2m
lagm(19) = lagsclm
lagm(20) = lagsc2m
lagm(21) = lainm
lagm(22) = lawam
lagm(23) = laflm
lagm(24) = lacem
lagm(25) = laexm
lagm(26) = lalim
lagm(27) = lamem
lagm(28) = laelm
lagm(29) = lagram

FOR I = 1 TO 29
IF lagm(I) - INT(lagm(I)) <> 0 THEN
lagm(I) = INT(lagm(I)) + 1
END IF
NEXT

END SUB
```

```
SUB ratio1 (s1, s2, s162, s188, s465, f1, f2, f3, f4, costra,
ib219, ib217, b219, b217, s1m, s2m, s162m, s188m, s465m, f1m,
f2m, f3m, f4m, c1, ib219m, ib217m, b219m, b217m, DIFF(),
ratio(), indact(), indactm(), dir()) STATIC

IF DIFF(1) <= .1 THEN
ratio(1) = 0
ELSE
ratio(1) = ((s1m - s1)) / DIFF(1)
END IF
dir(1) = s1
IF DIFF(2) <= .1 THEN
ratio(2) = 0
ELSE
ratio(2) = ((s2m - s2)) / DIFF(2)
END IF
dir(2) = s2
IF DIFF(3) <= .1 THEN
ratio(3) = 0
ELSE
ratio(3) = ((s162m - s162)) / DIFF(3)
END IF
dir(3) = s162
IF DIFF(4) <= .1 THEN
ratio(4) = 0
ELSE
ratio(4) = ((s188m - s188)) / DIFF(4)
END IF
dir(4) = s188
IF DIFF(5) <= .1 THEN
ratio(5) = 0
ELSE
ratio(5) = ((s465m - s465)) / DIFF(5)
END IF
dir(5) = s465
IF DIFF(6) <= .1 THEN
ratio(6) = 0
ELSE
ratio(6) = ((f1m - f1)) / DIFF(6)
END IF
dir(6) = f1
IF DIFF(7) <= .1 THEN
ratio(7) = 0
ELSE
ratio(7) = ((f2m - f2)) / DIFF(7)
END IF
dir(7) = f2
IF DIFF(8) <= .1 THEN
ratio(8) = 0
ELSE
ratio(8) = ((f3m - f3)) / DIFF(8)
END IF
dir(8) = f3
IF DIFF(9) <= .1 THEN
ratio(9) = 0
ELSE
ratio(9) = ((f4m - f4)) / DIFF(9)
END IF
dir(9) = f4
```

```
IF DIFF(10) <= .1 THEN
ratio(10) = 0
ELSE
ratio(10) = ((c1 - costra)) / DIFF(10)
END IF
dir(10) = c1
IF DIFF(11) <= .1 THEN
ratio(11) = 0
ELSE
ratio(11) = ((ib219m - ib219)) / DIFF(11)
END IF
dir(11) = ib219
IF DIFF(12) <= .1 THEN
ratio(12) = 0
ELSE
ratio(12) = ((ib217m - ib217)) / DIFF(12)
END IF
dir(12) = ib217
IF DIFF(13) <= .1 THEN
ratio(13) = 0
ELSE
ratio(13) = ((b219m - b219)) / DIFF(13)
END IF
dir(13) = b219
IF DIFF(14) <= .1 THEN
ratio(14) = 0
ELSE
ratio(14) = ((b217m - b217)) / DIFF(14)
END IF
dir(14) = b217
END SUB
```

```
SUB ratio2 (f6, f7, f8, f9, f10, s4, sem1, see1, tf374,
tf377, tf380, tf373, tf376, tf379, tf375, tf378, tf381, f6m,
f7m, f8m, f9m, f10m, s4m, sem1m, see1m, tf374m, tf377m,
tf380m, tf373m, tf376m, tf379m, tf375m, TF378M, tf381m,
DIFF(), ratio(), indact(), indactm(), dir(), nofloor) STATIC

IF DIFF(15) <= .1 THEN
ratio(15) = 0
ELSE
ratio(15) = ((f6m - f6)) / DIFF(15)
END IF
dir(15) = f6
IF DIFF(16) <= .1 THEN
ratio(16) = 0
ELSE
ratio(16) = ((f7m - f7)) / DIFF(16)
END IF
dir(16) = f7
IF DIFF(17) <= .1 THEN
ratio(17) = 0
ELSE
ratio(17) = ((f8m - f8)) / DIFF(17)
END IF
dir(17) = f8
IF DIFF(18) <= .1 THEN
ratio(18) = 0
ELSE
ratio(18) = ((f9m - f9)) / DIFF(18)
END IF
dir(18) = f9
IF DIFF(19) <= .1 THEN
ratio(19) = 0
ELSE
ratio(19) = ((f10m - f10)) / DIFF(19)
END IF
dir(19) = f10
IF DIFF(20) <= .1 THEN
ratio(20) = 0
ELSE
ratio(20) = ((s4m - s4)) / DIFF(20)
END IF
dir(20) = s4
IF DIFF(21) <= .1 THEN
ratio(21) = 0
ELSE
ratio(21) = ((sem1m - sem1)) / DIFF(21)
END IF
dir(21) = sem1
IF DIFF(22) <= .1 THEN
ratio(22) = 0
ELSE
ratio(22) = ((see1m - see1)) / DIFF(22)
END IF
dir(22) = see1
IF DIFF(23) <= .1 THEN
ratio(23) = 0
ELSE
ratio(23) = ((tf374m - tf374)) / DIFF(23)
END IF
```

```
dir(23) = tf374
IF DIFF(24) <= .1 THEN
ratio(24) = 0
ELSE
ratio(24) = ((tf377m - tf377)) / DIFF(24)
END IF
dir(24) = tf377
IF DIFF(25) <= .1 THEN
ratio(25) = 0
ELSE
ratio(25) = ((tf380m - tf380)) / DIFF(25)
END IF
dir(25) = tf380
IF DIFF(26) <= .1 THEN
ratio(26) = 0
ELSE
ratio(26) = ((tf373m - tf373)) / DIFF(26)
END IF
dir(26) = tf373
IF DIFF(27) <= .1 THEN
ratio(27) = 0
ELSE
ratio(27) = ((tf376m - tf376)) / DIFF(27)
END IF
dir(27) = tf376
IF DIFF(28) <= .1 THEN
ratio(28) = 0
ELSE
ratio(28) = ((tf379m - tf379)) / DIFF(28)
END IF
dir(28) = tf379
IF DIFF(29) <= .1 THEN
ratio(29) = 0
ELSE
ratio(29) = ((tf375m - tf375)) / DIFF(29)
END IF
dir(29) = tf375

IF DIFF(30) <= .1 THEN
ratio(30) = 0
ELSE
ratio(30) = ((TF378M - tf378)) / DIFF(30)
END IF
dir(30) = tf378
IF DIFF(31) <= .1 THEN
ratio(31) = 0
ELSE
ratio(31) = ((tf381m - tf381)) / DIFF(31)
END IF
dir(31) = tf381
FOR I = 1 TO nofloor - 1
'slab'
IF DIFF(23) <= .1 THEN
ratio(26 + (I) * 6) = 0
ELSE
ratio(26 + (I) * 6) = ((tf374m - tf374)) / DIFF(23)
END IF
```

```
IF DIFF(24) <= .1 THEN
ratio(27 + (I) * 6) = 0
ELSE
ratio(27 + I * 6) = ((tf377m - tf377)) / DIFF(24)
END IF
IF DIFF(25) <= .1 THEN
ratio(28 + (I) * 6) = 0
ELSE
ratio(28 + I * 6) = ((tf380m - tf380)) / DIFF(25)
END IF
'COLUMN'
IF DIFF(27) <= .1 THEN
ratio(30 + (I) * 6) = 0
ELSE
ratio(30 + I * 6) = ((tf376m - tf376)) / DIFF(27)
END IF
IF DIFF(26) <= .1 THEN
ratio(29 + (I) * 6) = 0
ELSE
ratio(29 + I * 6) = ((tf373m - tf373)) / DIFF(26)
END IF
IF DIFF(28) <= .1 THEN
ratio(31 + (I) * 6) = 0
ELSE
ratio(31 + I * 6) = ((tf379m - tf379)) / DIFF(28)
END IF
NEXT
'ROOF SLAB '
IF DIFF(23) <= .1 THEN
ratio(26 + nofloor * 6) = 0
ELSE
ratio(26 + nofloor * 6) = ((tf374m - tf374)) / DIFF(23)
END IF

'brick/block, finishes,service'

'STAIRCASES'
FOR I = 1 TO nofloor - 2
IF DIFF(29) <= .1 THEN
ratio(26 + nofloor * 6 + I) = 0
ELSE
ratio(26 + nofloor * 6 + I) = ((tf375m - tf375)) / DIFF(29)
END IF
IF DIFF(30) <= .1 THEN
ratio(27 + nofloor * 6 + I) = 0
ELSE
ratio(27 + nofloor * 6 + I) = ((TF378M - tf378)) / DIFF(30)
END IF
IF DIFF(31) <= .1 THEN
ratio(28 + nofloor * 6 + I) = 0
ELSE
ratio(28 + nofloor * 6 + I) = ((tf381m - tf381)) / DIFF(31)
END IF
NEXT
END SUB
```

```
SUB xyy (dur(), lag(), X(), Y(), nofloor, NOF, NOF1, PR)
STATIC

X(1) = 0
Y(1) = dur(1)
X(2) = X(1) + lag(1)
X(3) = X(2) + lag(2)
X(4) = X(2) + lag(3)
X(5) = X(2) + lag(4)
FOR I = 6 TO 9
X(I) = X(I - 1) + lag(I - 1)
Y(I) = X(I) + dur(I)
NEXT

FOR I = 2 TO 9
Y(I) = X(I) + dur(I)
NEXT

X(23) = X(9) + lag(9)
X(24) = X(23) + lag(17)
X(25) = X(24) + lag(18)
X(27) = X(23) + lag(10)
X(26) = X(27) + lag(15)
X(28) = X(26) + lag(16)
X(29) = X(23) + lag(12)
X(30) = X(29) + lag(19)
X(31) = X(30) + lag(20)
X(29) = X(23) + lag(12)
X(30) = X(29) + lag(19)
X(31) = X(30) + lag(20)


'SLABS'
FOR I = 1 TO nofloor - 1
X(26 + (I) * 6) = X(27) + lag(11) * I + lag(10) * (I - 1)
Y(26 + (I) * 6) = X(26 + I * 6) + dur(23)
X(27 + I * 6) = X(26 + I * 6) + lag(17)
Y(27 + (I) * 6) = X(27 + I * 6) + dur(24)
X(28 + I * 6) = X(27 + I * 6) + lag(18)
Y(28 + (I) * 6) = X(28 + I * 6) + dur(25)

'COLUMN'
X(30 + I * 6) = X(26 + 6 * I) + lag(10)
Y(30 + (I) * 6) = X(30 + I * 6) + dur(27)
X(29 + I * 6) = X(30 + I * 6) + lag(15)
Y(29 + (I) * 6) = X(29 + I * 6) + dur(26)
X(31 + I * 6) = X(29 + I * 6) + lag(16)
Y(31 + (I) * 6) = X(31 + I * 6) + dur(28)
NEXT

'ROOF SLAB AND ASPHALT'
X(26 + nofloor * 6) = X(27) + lag(11) * nofloor + lag(10) *
(nofloor - 1)
Y(26 + nofloor * 6) = X(26 + nofloor * 6) + dur(23)
X(10) = X(26 + (nofloor * 6)) + lag(29)
Y(10) = X(10) + dur(10)

'brick/block, finishes,service'
IF NOF = 1 THEN
```

```
X(13) = X(23) + lag(14)
X(14) = X(23) + lag(14)
ELSE
X(13) = X(26 + (NOF - 1) * 6) + lag(14)
X(14) = X(26 + (NOF - 1) * 6) + lag(14)
END IF
X(15) = X(13) + lag(21)
X(16) = X(13) + lag(22)
X(17) = X(13) + lag(23)
X(18) = X(13) + lag(24)
X(19) = X(13) + lag(25)
X(20) = X(13) + lag(26)
X(21) = X(13) + lag(27)
X(22) = X(13) + lag(28)
FOR I = 11 TO 31
Y(I) = X(I) + dur(I)
NEXT

'STAIRCASES'
I = 1
X(26 + nofloor * 6 + I) = Y(31)
Y(26 + nofloor * 6 + I) = X(26 + nofloor * 6 + I) + dur(29)
X(27 + nofloor * 6 + I) = X(26 + nofloor * 6 + I) + lag(19)
Y(27 + nofloor * 6 + I) = X(27 + nofloor * 6 + I) + dur(30)
X(28 + nofloor * 6 + I) = X(27 + nofloor * 6 + I) + lag(20)
Y(28 + nofloor * 6 + I) = X(28 + nofloor * 6 + I) + dur(31)
FOR I = 2 TO nofloor - 2
X(28 + nofloor * 6 + 3 * I - 4) = Y(28 + nofloor * 6 + 3 * I
- 5)
Y(28 + nofloor * 6 + 3 * I - 4) = X(28 + nofloor * 6 + 3 * I
- 4) + dur(29)
X(29 + nofloor * 6 + 3 * I - 4) = X(28 + nofloor * 6 + 3 * I
- 4) + lag(19)
Y(29 + nofloor * 6 + 3 * I - 4) = X(29 + nofloor * 6 + 3 * I
- 4) + dur(30)
X(30 + nofloor * 6 + 3 * I - 4) = X(29 + nofloor * 6 + 3 * I
- 4) + lag(20)
Y(30 + nofloor * 6 + 3 * I - 4) = X(30 + nofloor * 6 + 3 * I
- 4) + dur(31)
NEXT

'STAIRCASE-INT. BLOCK-BRICK'
IF NOF1 = 1 THEN
X(11) = X(29) + lag(13)
X(12) = X(29) + lag(13)
Y(11) = X(11) + dur(11)
Y(12) = X(12) + dur(12)
ELSE
X(11) = X(26 + nofloor * 6 + NOF1 - 1) + lag(13)
X(12) = X(26 + nofloor * 6 + NOF1 - 1) + lag(13)
Y(11) = X(11) + dur(11)
Y(12) = X(12) + dur(12)
END IF
FOR I = 1 TO 9 * nofloor + 21
IF Y(I) > PR THEN
PR = Y(I)
END IF
NEXT
END SUB
```

```
SUB XYYM (DURM(), lagm(), XM(), YM(), nofloor, NOFm, NOF1m,
prM) STATIC
XM(1) = 0
YM(1) = DURM(1)
XM(2) = XM(1) + lagm(1)
XM(3) = XM(2) + lagm(2)
XM(4) = XM(2) + lagm(3)
XM(5) = XM(2) + lagm(4)
FOR I = 6 TO 9
XM(I) = XM(I - 1) + lagm(I - 1)
YM(I) = XM(I) + DURM(I)
NEXT

FOR I = 2 TO 9
YM(I) = XM(I) + DURM(I)
NEXT

XM(23) = XM(9) + lagm(9)
XM(24) = XM(23) + lagm(17)
XM(25) = XM(24) + lagm(18)
XM(27) = XM(23) + lagm(10)
XM(26) = XM(27) + lagm(15)
XM(28) = XM(26) + lagm(16)
XM(29) = XM(23) + lagm(12)
XM(30) = XM(29) + lagm(19)
XM(31) = XM(30) + lagm(20)
XM(29) = XM(23) + lagm(12)
XM(30) = XM(29) + lagm(19)
XM(31) = XM(30) + lagm(20)

'SLABS'
FOR I = 1 TO nofloor - 1
XM(26 + (I) * 6) = XM(27) + lagm(11) * I + lagm(10) * (I - 1)
YM(26 + I * 6) = XM(26 + I * 6) + DURM(23)
XM(27 + I * 6) = XM(26 + I * 6) + lagm(17)
YM(27 + I * 6) = XM(27 + I * 6) + DURM(24)
XM(28 + I * 6) = XM(27 + I * 6) + lagm(18)
YM(28 + I * 6) = XM(28 + I * 6) + DURM(25)

'COLUMN'
XM(30 + I * 6) = XM(26 + 6 * I) + lagm(10)
YM(30 + I * 6) = XM(30 + I * 6) + DURM(27)
XM(29 + I * 6) = XM(30 + I * 6) + lagm(15)
YM(29 + I * 6) = XM(29 + I * 6) + DURM(26)
XM(31 + I * 6) = XM(29 + I * 6) + lagm(16)
YM(31 + I * 6) = XM(31 + I * 6) + DURM(28)
NEXT

'ROOF SLAB AND ASPHALT'
XM(26 + nofloor * 6) = XM(27) + lagm(11) * nofloor + lagm(10)
* (nofloor - 1)
YM(26 + nofloor * 6) = XM(26 + nofloor * 6) + DURM(23)
XM(10) = XM(26 + (nofloor * 6)) + lagm(29)
YM(10) = XM(10) + DURM(10)

'brick/block, finishes,service'
IF NOFm = 1 THEN
XM(13) = XM(23) + lagm(14)
XM(14) = XM(23) + lagm(14)
```

```
ELSE
XM(13) = XM(26 + (NOFm - 1) * 6) + lagm(14)
XM(14) = XM(26 + (NOFm - 1) * 6) + lagm(14)
END IF
XM(15) = XM(13) + lagm(21)
XM(16) = XM(13) + lagm(22)
XM(17) = XM(13) + lagm(23)
XM(18) = XM(13) + lagm(24)
XM(19) = XM(13) + lagm(25)
XM(20) = XM(13) + lagm(26)
XM(21) = XM(13) + lagm(27)
XM(22) = XM(13) + lagm(28)

FOR I = 11 TO 31
YM(I) = XM(I) + DURM(I)
NEXT

'STAIRCASES'
I = 1
XM(26 + nofloor * 6 + I) = YM(31)
YM(26 + nofloor * 6 + I) = XM(26 + nofloor * 6 + I) +
DURM(29)
XM(27 + nofloor * 6 + I) = XM(26 + nofloor * 6 + I) +
lagm(19)
YM(27 + nofloor * 6 + I) = XM(27 + nofloor * 6 + I) +
DURM(30)
XM(28 + nofloor * 6 + I) = XM(27 + nofloor * 6 + I) +
lagm(20)
YM(28 + nofloor * 6 + I) = XM(28 + nofloor * 6 + I) +
DURM(31)
FOR I = 2 TO (nofloor - 2)
XM(28 + nofloor * 6 + 3 * I - 4) = YM(28 + nofloor * 6 + 3 *
I - 4 - 1)
YM(28 + nofloor * 6 + 3 * I - 4) = XM(28 + nofloor * 6 + 3 *
I - 4) + DURM(29)
XM(29 + nofloor * 6 + 3 * I - 4) = XM(28 + nofloor * 6 + 3 *
I - 4) + lagm(19)
YM(29 + nofloor * 6 + 3 * I - 4) = XM(29 + nofloor * 6 + 3 *
I - 4) + DURM(30)
XM(30 + nofloor * 6 + 3 * I - 4) = XM(29 + nofloor * 6 + 3 *
I - 4) + lagm(20)
YM(30 + nofloor * 6 + 3 * I - 4) = XM(30 + nofloor * 6 + 3 *
I - 4) + DURM(31)
NEXT

'STAIRCASE-INT. BLOCK-BRICK'
IF NOF1m = 1 THEN
XM(11) = XM(29) + lagm(13)
XM(12) = XM(29) + lagm(13)
YM(11) = XM(11) + DURM(11)
YM(12) = XM(12) + DURM(12)
ELSE
XM(11) = XM(26 + nofloor * 6 + NOF1m - 1) + lagm(13)
XM(12) = XM(26 + nofloor * 6 + NOF1m - 1) + lagm(13)
YM(11) = XM(11) + DURM(11)
YM(12) = XM(12) + DURM(12)
END IF
```

```
FOR I = 1 TO 9 * nofloor + 21
IF YM(I) > prM THEN
prM = YM(I)
END IF
NEXT
END SUB
```

```
DECLARE SUB constr1 (DUR!(), durm!(), lag!(),  nofloor!,
nof1m!, stairdaym!, nofm!)
DECLARE SUB constr2 (lag!(), DUR!(), projdur!, nofloor!,
PROJDURM!)
DECLARE SUB constn (DUR!(), durm!(), lag!(), nofloor!)
DECLARE SUB constrt (DUR!(), durm!(), nofloor!)
DECLARE SUB constrfs (lag(), nofloor!, DUR())
DIM lag(29), durm(31), DUR(31)


OPEN "new.dat" FOR INPUT AS #1
INPUT #1, new
CLOSE 1
IF new = 1 THEN
new = 0
neww = 0
END IF
OPEN "new.dat" FOR OUTPUT AS #1
PRINT #1, new
PRINT #1, neww
CLOSE 1

OPEN "NOOO.DAT" FOR INPUT AS #8
INPUT #8, nofloor
CLOSE #8
OPEN "OPT4.dat" FOR INPUT AS #10
INPUT #10, projdur, PROJDURM
CLOSE #10
OPEN "coptsif1.dat" FOR INPUT AS #5
INPUT #5, dummy1, dummy2, stairday, dummy3, dummy4, stairdaym
CLOSE #5
OPEN "nof.dat" FOR INPUT AS #11
INPUT #11, nof, nof1, nofm, nof1m
CLOSE #11
ss = 0
pp = 1
OPEN "constr1.dat" FOR OUTPUT AS #1
OPEN "constr2.dat" FOR OUTPUT AS #2
OPEN "constr3.dat" FOR OUTPUT AS #3
OPEN "constr4.dat" FOR OUTPUT AS #4
OPEN "constr5.dat" FOR OUTPUT AS #5

OPEN "lagaa.dat" FOR INPUT AS #19
'lags for the mimimum duration'

FOR i = 1 TO 29
INPUT #19, lag(i)
NEXT
CLOSE 19

OPEN "durn.dat" FOR INPUT AS #19
FOR i = 1 TO 31
INPUT #19, DUR(i)
NEXT
CLOSE 19

OPEN "durm.dat" FOR INPUT AS #20
```

```
FOR i = 1 TO 31
INPUT #20, durm(i)
NEXT
CLOSE 20


CALL constr1(DUR(), durm(), lag(), nofloor, nof1m,
stairdaym, nofm)


CALL constr2(lag(), DUR(), projdur, nofloor, PROJDURM)

CALL constrfs(lag(), nofloor, DUR())

CALL constrt(DUR(), durm(), nofloor)

CALL constn(DUR(), durm(), lag(), nofloor)

CLOSE 1, 2, 3, 4, 5, 6, 7, 8
```

```
SUB constn (DUR(), durm(), lag(), nofloor)

'SLABS AND COLUMNS'
aa = 6 * nofloor - 5
bb = aa * 2
cc = 6 * (nofloor - 2)

T = 1
PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 27
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 29 TO 32 + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 34 + (6 * (T - 1)) TO 58 + aa
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 60 + aa TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

FOR T = 2 TO nofloor - 1
PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 31 + 6 * (T - 1)
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0; 0; 0;
FOR i = 36 + 6 * (T - 1) TO 61 + 6 * (T - 1)
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 63 + 6 * (T - 1) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0
NEXT

FOR T = 1 TO nofloor - 1
PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 32 + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0;
FOR i = 35 + (6 * (T - 1)) TO 63 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 65 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0
```

```
PRINT #5, "le";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 63 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 65 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 33 + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0;
FOR i = 36 + (6 * (T - 1)) TO 64 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 66 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0


PRINT #5, "le";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 64 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 66 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0



'COL/ WALL'

PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 32 + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0; 0; 0; 0;
FOR i = 38 + (6 * (T - 1)) TO 63 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 65 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

PRINT #5, "le";
```

```
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 63 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 65 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0


PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 35 + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0;
FOR i = 38 + (6 * (T - 1)) TO 66 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 68 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

PRINT #5, "le";
PRINT #5, USING "###.#"; DUR(27) - durm(27);
FOR i = 1 TO 66 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 1;
FOR i = 68 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;


FOR i = 1 TO 35 + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0; 0;
FOR i = 39 + (6 * (T - 1)) TO 65 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 67 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

PRINT #5, "le";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 65 + aa + (6 * (T - 1))
PRINT #5, USING "###.#"; 0;
NEXT
```

```
PRINT #5, USING "###.#"; 0
FOR i = 67 + aa + (6 * (T - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0
NEXT

'ROOF SLAB'

PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR i = 1 TO 36 + (6 * (nofloor - 1 - 1))
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0; 0; 0;
FOR i = 41 + (6 * (nofloor - 1) - 1) TO 66 + aa + (6 *
(nofloor - 1 - 1) + 1)
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR i = 66 + aa + (6 * (nofloor - 1 - 1)) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

PRINT #5, "le";
PRINT #5, USING "###.#"; DUR(27) - durm(27);
FOR i = 1 TO 66 + aa + (6 * (nofloor - 1 - 1) + 1)
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 1;
FOR i = 66 + aa + (6 * (nofloor - 1) - 1) TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

'STAIRCASES'
FOR i = 1 TO nofloor - 2
IF i = 1 THEN
PRINT #5, "ge";
PRINT #5, USING "###.#"; DUR(31);
FOR j = 1 TO 31
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; -1;
FOR j = 33 TO 62 + aa
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 1;
FOR j = 64 + aa TO 63 + bb
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 1;
FOR j = 65 + bb TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

ELSE
```

```
PRINT #5, "ge";
PRINT #5, USING "###.#"; DUR(31);
FOR j = 1 TO 62 + bb + (i - 1) * 3
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; -1; 1;
FOR j = 65 + bb + (i - 1) * 3 TO 65 + bb + cc /2 +(i - 1)* 3
PRINT #5, USING "###.#"; 0;
NEXT
IF i = nofloor - 2 THEN
PRINT #5, USING "###.#"; 1
ELSE
PRINT #5, USING "###.#"; 1;
FOR j = 67 + bb + cc / 2 + (i - 1) * 3 TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0
END IF
END IF

PRINT #5, "le";
PRINT #5, USING "###.#"; DUR(31) - durm(31);
FOR j = 1 TO 65 + bb + cc / 2 + (i - 1) * 3
PRINT #5, USING "###.#"; 0;
NEXT
IF i = nofloor - 2 THEN
PRINT #5, USING "###.#"; 1
ELSE
PRINT #5, USING "###.#"; 1;
FOR j = 67 + bb + cc / 2 + (i - 1) * 3 TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0
END IF

PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR j = 1 TO 63 + bb + (i - 1) * 3
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0;
FOR j = 66 + bb + (i - 1) * 3 TO 63 + bb + cc / 2 +(i - 1)*3
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR j = 65 + bb + cc / 2 + (i - 1) * 3 TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

PRINT #5, "le";
PRINT #5, USING "###.#"; DUR(29) - durm(29);
FOR j = 1 TO 63 + bb + cc / 2 + (i - 1) * 3
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 1;
FOR j = 65 + bb + cc / 2 + (i - 1) * 3 TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
```

```
PRINT #5, USING "###.#"; 0

PRINT #5, "ge";
PRINT #5, USING "###.#"; 0;
FOR j = 1 TO 64 + bb + (i - 1) * 3
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0; 0;
FOR j = 67 + bb + (i - 1) * 3 TO 64 + bb + cc / 2 +(i - 1)*3
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0;
FOR j = 66 + bb + cc / 2 + (i - 1) * 3 TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0

PRINT #5, "le";
PRINT #5, USING "###.#"; DUR(30) - durm(30);
FOR j = 1 TO 64 + bb + cc / 2 + (i - 1) * 3
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 1;
FOR j = 66 + bb + cc / 2 + (i - 1) * 3 TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0
NEXT

PRINT #5, "le";
PRINT #5, USING "###.#"; DUR(30) - durm(30);
FOR j = 1 TO 65 + bb + cc / 2 + (i - 1) * 3
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 1;
FOR j = 67 + bb + cc / 2 + (i - 1) * 3 TO 62 + bb + cc
PRINT #5, USING "###.#"; 0;
NEXT
PRINT #5, USING "###.#"; 0
END SUB
```

```
SUB constr1 (DUR(), durm(), lag(), nofloor, noflm,
stairdaym, nofm) STATIC
aa = 6 * nofloor - 5
bb = aa * 2
cc = 6 * (nofloor - 2)

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(1);
PRINT #1, USING "###.#"; 0; -1; 1;
FOR i = 4 TO 32 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 34 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(2);
PRINT #1, USING "###.#"; 0; 0; -1; 1;
FOR i = 5 TO 33 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 35 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
PRINT #1, USING "###.#"; 0; 0; 0; -1; 1;
FOR i = 6 TO 34 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 36 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "eq";
PRINT #1, USING "###.#"; 0;
PRINT #1, USING "###.#"; 0; 0; 0; 0; -1; 1;
FOR i = 7 TO 35 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 37 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

'additional'
PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
PRINT #1, USING "###.#"; 0; 0; 0; 0; 0; 0; -1; 1;
```

```
FOR i = 8 TO 35 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 37 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(5);
PRINT #1, USING "###.#"; 0; 0; 0; 0; 0; -1; 1;
FOR i = 8 TO 36 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 38 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

'additional'
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(5);
PRINT #1, USING "###.#"; 0; 0; 0; 0; 0; -1; 1;
FOR i = 8 TO 36 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 36 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(6);
PRINT #1, USING "###.#"; 0; 0; 0; 0; 0; 0; -1; 1;
FOR i = 9 TO 37 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 39 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(7);
PRINT #1, USING "###.#"; 0; 0; 0; 0; 0; 0; 0; -1; 1;
FOR i = 10 TO 38 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 40 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(8);
PRINT #1, USING "###.#"; 0; 0; 0; 0; 0; 0; 0; 0; -1; 1;
```

```
FOR i = 11 TO 39 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 41 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(9);
PRINT #1, USING "###.#"; 0; 0; 0; 0; 0; 0; 0; 0; 0; -1;
FOR i = 11 TO 23
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 25 TO 40 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 42 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
FOR i = 1 TO 23
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0;
FOR i = 26 TO 54 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 56 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
FOR i = 1 TO 24
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0;
FOR i = 27 TO 55 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 57 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
```

```
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(10);
FOR i = 1 TO 23
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1; 0; 0; 0; 1;
FOR i = 29 TO 54 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 56 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
FOR i = 1 TO 26
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0;
FOR i = 29 TO 58 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 60 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
FOR i = 1 TO 26
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0; 0;
FOR i = 30 TO 57 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 59 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

'slab (1)'
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(11);
FOR i = 1 TO 27
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1; 0; 0; 0; 0; 1;
FOR i = 34 TO 59 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 61 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
```

```
PRINT #1, USING "###.#"; 0

PRINT #1, "le";
PRINT #1, USING "###.#"; DUR(27) - durm(27);
FOR i = 1 TO 27
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0; 0; 0; 0; 0;
FOR i = 34 TO 59 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 61 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

'COL (1-2)-SLAB(2)'

FOR ii = 1 TO nofloor - 2
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(11);
FOR i = 1 TO 6 * ii + 30
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1; 0; 1;
FOR i = 6 * ii + 34 TO 6 * ii + 62 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 6 * ii + 64 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
NEXT

FOR ii = 1 TO nofloor - 2
PRINT #1, "le";
PRINT #1, USING "###.#"; DUR(27) - durm(27);
FOR i = 1 TO 6 * ii + 62 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 6 * ii + 64 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
NEXT

FOR ii = 1 TO nofloor - 1
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(10);
FOR i = 1 TO 6 * (ii - 1) + 32
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1; 0; 0; 0; 1;
FOR i = 6 * (ii - 1) + 38 TO 6 * (ii - 1) + 64 + aa
PRINT #1, USING "###.#"; 0;
NEXT
```

```
PRINT #1, USING "###.#"; 1;
FOR i = 6 * (ii - 1) + 66 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
NEXT

FOR ii = 1 TO nofloor - 1
PRINT #1, "le";
PRINT #1, USING "###.#"; DUR(23) - durm(23);
FOR i = 1 TO 6 * (ii - 1) + 64 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 6 * (ii - 1) + 66 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
NEXT

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(12);
FOR i = 1 TO 29
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1; 0; 0; -1;
FOR i = 34 TO 32 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 34 + aa TO 65 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 67 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
FOR i = 1 TO 29
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0;
FOR i = 32 TO 60 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0;
FOR i = 63 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
FOR i = 1 TO 30
PRINT #1, USING "###.#"; 0;
NEXT
```

```
PRINT #1, USING "###.#"; 0; 0;
FOR i = 33 TO 61 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 63 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(29);
FOR i = 1 TO 10
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 12 TO 31 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 33 + aa TO 62 + bb
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 64 + bb TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "le";
PRINT #1, USING "###.#"; DUR(23) - durm(23);
FOR i = 1 TO 62 + bb
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 64 + bb TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(11);
FOR i = 1 TO 29 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1; 0; 1;
FOR i = 33 + aa TO 60 + bb
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 62 + bb TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "le";
PRINT #1, USING "###.#"; DUR(27) - durm(27);
```

```
FOR i = 1 TO 29 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0; 0;
FOR i = 33 + aa TO 60 + bb
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 62 + bb TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

IF nof1m = 1 THEN
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(13);
FOR i = 1 TO 11
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 13 TO 29
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 31 TO 60 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1; 0;
FOR i = 63 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(13);
FOR i = 1 TO 12
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 14 TO 29
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 31 TO 60 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 62 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

ELSE

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(13);
FOR i = 1 TO 11
PRINT #1, USING "###.#"; 0;
NEXT
```

```
PRINT #1, USING "###.#"; 1;
FOR i = 13 TO 29
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 31 TO 60 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0;
FOR i = 63 + aa TO 63 + bb + (nofm - 2) * 3
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 65 + bb + (nofm - 2) * 3 TO 62 + bb + cc + 1 - 3 *
(nofloor - 2) + 3 * (nofm - 2)
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 65 + bb + cc - (3 * nofloor - 2) + 3 * (nofm - 2) TO
62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(13);
FOR i = 1 TO 12
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 14 TO 29
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 31 TO 60 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0; 0;
FOR i = 63 + aa TO 63 + bb + (nofm - 2) * 3
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 65 + bb + (nofm - 2) * 3 TO 62 + bb + cc + 1 - 3 *
(nofloor - 2) + 3 * (nofm - 2)
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 65 + bb + cc - 3 * (nofloor - 2) + 3 * (nofm - 2) TO
62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
END IF

IF nofm = 0 THEN
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(14);
FOR i = 1 TO 13
PRINT #1, USING "###.#"; 0;
```

```
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 15 TO 23
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 25 TO 54 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 56 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(14);
FOR i = 1 TO 14
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 16 TO 23
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 25 TO 54 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 56 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

ELSE
PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(14);
FOR i = 1 TO 13
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 15 TO 26 + nofm * 6
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 28 + nofm * 6 TO 58 + bb
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 60 + bb TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0

PRINT #1, "ge";
PRINT #1, USING "###.#"; lag(14);
FOR i = 1 TO 14
PRINT #1, USING "###.#"; 0;
NEXT
```

```
PRINT #1, USING "###.#"; 1;
FOR i = 16 TO 26 + nofm * 6
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; -1;
FOR i = 28 + nofm * 6 TO 58 + bb
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 1;
FOR i = 60 + bb TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
END IF

PRINT #1, "ge";
PRINT #1, USING "###.#"; 0;
FOR i = 1 TO 14
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 16 TO 23
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 25 TO 54 + aa
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0;
FOR i = 56 + aa TO 62 + bb + cc
PRINT #1, USING "###.#"; 0;
NEXT
PRINT #1, USING "###.#"; 0
END SUB
```

```
SUB constr2 (lag(), DUR(), projdur, nofloor, PROJDURM)
STATIC
'Xp'
aa = 6 * nofloor - 5
bb = aa * 2
cc = 6 * (nofloor - 2)
PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(10);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 10
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 12 TO 41 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 43 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(11);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 11
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 13 TO 42 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 44 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(12);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 12
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 14 TO 43 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 45 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(15);
PRINT #2, USING "###.#"; 1;
```

```
FOR i = 2 TO 15
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 17 TO 46 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 48 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(16);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 16
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 18 TO 47 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 49 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(17);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 17
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 19 TO 48 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 50 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(18);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 18
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 20 TO 49 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 51 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
```

```
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(19);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 19
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 21 TO 50 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 52 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(20);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 20
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 22 TO 51 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 53 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(21);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 21
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
FOR i = 23 TO 52 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 54 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; DUR(22);
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 22
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; -1;
```

```
FOR i = 24 TO 53 + aa
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 1;
FOR i = 55 + aa TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "ge";
PRINT #2, USING "###.#"; PROJDURM;
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "le";
PRINT #2, USING "###.#"; projdur;
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

PRINT #2, "le";
PRINT #2, USING "###.#"; projdur;
PRINT #2, USING "###.#"; 1;
FOR i = 2 TO 62 + bb + cc
PRINT #2, USING "###.#"; 0;
NEXT
PRINT #2, USING "###.#"; 0

END SUB
```

```
SUB constrfs (lag(), nofloor, DUR()) STATIC
aa = 6 * nofloor - 5
bb = aa * 2
cc = 6 * (nofloor - 2)
'finish service (ge)'
PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(21);
FOR i = 1 TO 13
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 1;
FOR i = 17 TO 44 + aa
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(22);
FOR i = 1 TO 13
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 0; 1;
FOR i = 18 TO 44 + aa
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(23);
FOR i = 1 TO 13
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 0; 0; 1;
FOR i = 19 TO 44 + aa
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(24);
FOR i = 1 TO 13
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 0; 0; 0; 1;
FOR i = 20 TO 44 + aa
PRINT #3, USING "###.#"; 0;
NEXT
```

```
PRINT #3, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(25);
FOR i = 1 TO 13
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 0; 0; 0; 0; 1;
FOR i = 21 TO 44 + aa
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(26);
FOR i = 1 TO 13
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 0; 0; 0; 0; 0; 1;
FOR i = 22 TO 44 + aa
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(27);
FOR i = 1 TO 13
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 0; 0; 0; 0; 0; 0; 1;
FOR i = 23 TO 44 + aa
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(28);
FOR i = 1 TO 13
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 0; 0; 0; 0; 0; 0; 0; 1;
```

```
FOR i = 24 TO 44 + aa
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

PRINT #3, "ge";
PRINT #3, USING "###.#"; lag(28);
FOR i = 1 TO 14
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; -1; 0; 0; 0; 0; 0; 0; 0; 1;
FOR i = 24 TO 45 + aa
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 1;
FOR i = 47 + aa TO 62 + bb + cc
PRINT #3, USING "###.#"; 0;
NEXT
PRINT #3, USING "###.#"; 0

END SUB
```

```
SUB constrt (DUR(), durm(), nofloor) STATIC
aa = 6 * nofloor - 5
bb = aa * 2
cc = 6 * (nofloor - 2)
PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(1) - durm(1);
FOR i = 1 TO 32 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 34 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(2) - durm(2);
FOR i = 1 TO 33 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 35 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(3) - durm(3);
FOR i = 1 TO 34 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 36 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(4) - durm(4);
FOR i = 1 TO 35 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 37 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(5) - durm(5);
FOR i = 1 TO 36 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 38 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0
```

```
PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(6) - durm(6);
FOR i = 1 TO 37 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 39 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(7) - durm(7);
FOR i = 1 TO 38 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 40 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(8) - durm(8);
FOR i = 1 TO 39 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 41 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(9) - durm(9);
FOR i = 1 TO 40 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 42 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(10) - durm(10);
FOR i = 1 TO 41 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 43 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(11) - durm(11);
FOR i = 1 TO 42 + aa
PRINT #4, USING "###.#"; 0;
```

```
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 44 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(12) - durm(12);
FOR i = 1 TO 43 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 45 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(13) - durm(13);
FOR i = 1 TO 44 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 46 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(14) - durm(14);
FOR i = 1 TO 45 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 47 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(15) - durm(15);
FOR i = 1 TO 46 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 48 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(16) - durm(16);
FOR i = 1 TO 47 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
```

```
FOR i = 49 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(17) - durm(17);
FOR i = 1 TO 48 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 50 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(18) - durm(18);
FOR i = 1 TO 49 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 51 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(19) - durm(19);
FOR i = 1 TO 50 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 52 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(20) - durm(20);
FOR i = 1 TO 51 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 53 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(21) - durm(21);
FOR i = 1 TO 52 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 54 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0
```

```
PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(22) - durm(22);
FOR i = 1 TO 53 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 55 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(23) - durm(23);
FOR i = 1 TO 63 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 65 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; 0;
FOR i = 1 TO 55 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0;
FOR i = 57 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; 0;
FOR i = 1 TO 56 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0;
FOR i = 58 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; 0;
FOR i = 1 TO 57 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0;
FOR i = 59 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(27) - durm(27);
```

```
FOR i = 1 TO 58 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 60 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; 0;
FOR i = 1 TO 59 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0;
FOR i = 61 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(29) - durm(29);
FOR i = 1 TO 60 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1; 0;
FOR i = 63 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(30) - durm(30);
FOR i = 1 TO 61 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 63 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(31) - durm(31);
FOR i = 1 TO 62 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 64 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; 0;
FOR i = 1 TO 62 + aa
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0;
```

```
FOR i = 64 + aa TO 62 + bb + cc
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(23) - durm(23);
FOR i = 1 TO 62 + bb
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 1 TO cc - 1
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0

PRINT #4, "le";
PRINT #4, USING "###.#"; DUR(23) - durm(23);
FOR i = 1 TO 62 + bb
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 1;
FOR i = 1 TO cc - 1
PRINT #4, USING "###.#"; 0;
NEXT
PRINT #4, USING "###.#"; 0


END SUB
```

```
      dimension dur(31),nxi(500),iss(500,2),c(500,500),EE(500)
      real m,ggg,hindir,dirtot,dirtotm,new_bb
      real cost_increase,new_ind ,new_ind1,new_total,ind_dec
      double precision  B(1000),A(1000,1000),XX,xm,zcm,ck
      real d(1000),icj(1000),ratio(1000),mustif,old_dir
      real dir(31),cost_incr(32),tol,cv,nofloor1
      INTEGER t, k ,jm,n,ul,TA,new_b,gfa,f,tut,lm,nofloor
      INTEGER I1,ITER
      character*1  kode(500), emel(2)
      data emel/'1','g'/
      NPR=0
      cost_increase=0.0
      tol =0.00001
      cv=1000.0
      iter=0
      lm=0
      tut=0
      open (7,file='new.dat',status='old')
      read (7,*) new_b
      read(7,*) new_bb
      close (7)
      if (new_b .eq.0) then
      open (11,file=' projco.dat',status='old')
      write(11,*) 0
      write (11,*) 1
      close (11)
      open (8,file='data.dat',status='old')
      else
      open (8,file='data.dat',status='old')
      do 10 i=1,10000
      read (8,'(/)',end=999)
10    continue
999   backspace (unit=8)

      endif
      open (7,file='nooo.dat',status='old')
      read (7,*) nofloor1
      close  (7)
      nofloor=int(nofloor1)
      open (17,file='tott2.dat',status='old')
      read (17,*) pr
      read (17,*) prm
      read (17,*) dirtot
      read (17,*) dirtotm
      read (17,*) hindir
      close (17)
      kod=1
      knt=0
      aa=6*nofloor-5
      bb=2*aa
```

```
      cc=6*(nofloor-2)
 c    'n=number of variables, m=number of constraints'

      n=18*NOFLOOR+41
      m=17*NOFLOOR+58+4*nofloor-8+3+3
      murti=4*nofloor-8+3+3

      open (22,file='opt2.dat',status='old')
522   format (f15.2/)
      do 709 jj=1,n
      backspace (unit=22,iostat=i,err=1901)
      read (22,522) d(jj)
      ratio(jj)=d(jj)
709   continue
1901  close (22)
      close(24)
      close (23)
      n1=n
      nn=n
      m1=m+1

      open(10,file='constr1.dat',status='old',form='formatted')
      open (23,file='dd.dat',status='old',form='formatted')
33    format (50(a1,1x,f5.1/))
      do 811 k=2,26+murti
      kk=k
      backspace (unit=10,iostat=i,err=8901)
      read (10,33) kode(kk),ggg
      b(kk)=ggg
      write (23,33) kode(kk),b(kk)
811   continue
      close (10)

8901  open(10,file='constr1.dat',status='old',form='formatted')
34    format (50(7x,500f5.1/))
      backspace (unit=10,iostat=i,err=901)
      do 9011 k=2,26+murti
      read(10,34)(c(k,jj),jj=1,N)
9011  continue
901   if (i.gt.26+murti) then
      close (10)
      endif

      open (12,file='constr3.dat',status='old',form='formatted')
      do 815 mu=1,8
      mku=mu+26+murti
      backspace (unit=12,iostat=i,err=3901)
      read (12,33) kode(mku),ggg
      b(mku)=ggg
815   continue
      close (12)

3901  open(12,file='constr3.dat',status='old',form='formatted')
64    format (50(7x,500f5.1/))
```

```
      backspace (unit=12,iostat=i,err=301)
      t=26+murti
      do 3011 ml=1,8
      read(12,64)(c(ml+t,jj),jj=1,N)
3011  continue
301   if (i.gt.8) then
      close (12)
      endif

      open (13,file='constr4.dat',status='old',form='formatted')
      do 818 uu=1,33
      muu=uu+34+murti
      backspace (unit=13,iostat=i,err=4901)
      read (13,33) kode(muu),ggg
      b(muu)=ggg

818   continue
4901  close (13)
      open (13,file='constr4.dat',status='old',form='formatted')
      backspace (unit=13,iostat=i,err=401)
      t=34+murti
      do 4011 ul=1,33
      read(13,64)(c(ul+t,jj),jj=1,N)
4011  continue
401   if (i.gt.33) then
      close (13)
      endif

      open (13,file='constr5.dat',status='old',form='formatted')
663   format (319(a1,1x,f5.1/))
664   format (319(7x,500f5.1/))
      do 8108 uuu=1,(17*NOFLOOR-21)
      mcu=uuu+67+murti
      backspace (unit=13,iostat=i,err=5901)
      read (13,663) kode(mcu),g1gg
      b(mcu)=g1gg

8108  continue
      close (13)
5901  open(13,file='constr5.dat',status='old',form='formatted')
      backspace (unit=13,iostat=i,err=17901)
      ta=67+murti
      do 54011 ul=1,(17*nofloor-21)
      read(13,664)(c(ul+ta,jj),jj=1,N)
54011 continue
17901 if (i.gt.((17*nofloor-21))) then
      close (13)
      endif

      open (11,file='constr2.dat',status='old',form='formatted')
      do 813 ku=1,13
      kku=ku+57+(17*NOFLOOR-21)+murti
      backspace (unit=11,iostat=i,err=2901)
      read (11,33) kode(kku),ggg
```

```
      b(kku)=ggg
      if ((new_b .eq. 1).and.(ku.eq.13)) then
      b(kku)=new_bb
c     'b(kku)=pr-ff'
      endif
813   continue

      close (11)
2901  open(11,file='constr2.dat',status='old',form='formatted')
54    format (50(7x,500f5.1/))
      backspace (unit=11,iostat=i,err=201)
      t=57+(17*NOFLOOR-21)+murti
      do 2011 kl=1,13
      read(11,54)(c(kl+t,jj),jj=1,N)
2011  continue
201   if (i.gt.13) then
      close (11)
      endif
      close(23)

      do 331 i=1,n
      i1=i+knt
      iss(i,1)=i1
      iss(i,2)=0
      icj(i1)=d(i)
      do 332 i2=2,m1
      a(i2,i1)=c(i2,i)
332   continue

331   continue

      do 410 i= 2,m1
      if (ichar(kode(i)) .ne.ichar (emel(2))) THEN
      goto 410
      ENDIF
      n=n+1
      a(i,n)=-1
410   continue
      av=0

      do 420 i=2,m1
      n=n+1
      a(i,n)=1
      nxi(i)=n
      if (ichar(kode(i)) .eq.ichar(emel(1))) THEN
      goto 420
      ENDIF
      av=1
      icj(n)=cv
420   continue

      print *,b(kku)
      iter=0
      if (iter .eq. 0) THEN
```

```
      goto 4444
      ENDIF

4444  n1=1
      n2=6

43    if ((n2-n).le.0) then
      goto 45
      endif
      n2=n

45    do 5500 j=n1,n2
      if((n2-n).ge.0) THEN
      goto 5500
      ENDIF
      n1=n1+6
      n2=n2+6
      goto 43
5500  continue

      if (npr .ne. 2) THEN
      goto 21
      ENDIF
      do 234 jj=1,nn
      d(jj)=0
      do 235 i=2,m1
      k=nxi(i)
      if (k .eq. iss(jj,1)) then
      d(jj)=b(i)
      endif
      if (k .eq. iss(jj,2)) then
      d(jj) =-b(i)
      ENDIF
235   continue
234   continue



      open (12,file=' durn.dat',status='old')
      do 1880 j=1, 31
      backspace (unit=12,iostat=i,err=19001)
      read(12,121)dur(j)
1880  continue
19001 close (12)
121   format (f9.2/)
      open (10,file='dir.dat',status='old')
      do 18800 j=1, 31
      backspace (unit=10,iostat=i,err=18801)
      read(10,121)dir(j)
18800 continue
18801 close (10)

      do 1982  i=1,62+bb+cc
      cost_incr(i)=ratio(i)*d(i)
```

```
      cost_increase=cost_increase+cost_incr(i)
1982  continue
      open (11,file=' cinc.dat',status='old')
      write(11,*) cost_increase
      do 1995 i=1,62+bb+cc
      write(11,*) cost_incr(i),dir(i),ratio(i), d(i)
1995  continue
      open (11,file=' projco.dat',status='old')
      read (11,*) old_total
      read (11,*) old_dir
      close (11)
      mustif=cost_increase-old_dir
      print *, mustif
      if (mustif .ge. 990.000 ) then
      open (2,file=' trial.dat',status='old')
      write (2,*) 1.0
      close (2)
      WRITE (8,*)'This project can not be accelerated more.'
      write (8,*)'                                          '
      close (8)
      goto 7777
      endif
      write (8,*) 'optimal solution:/objective function:',b(1)
      write (8,*)'project duration is:', d(1), 'days'
      new_ind=hindir/pr
      hh=d(1)-pr
      if (hh.lt.0.0) then
      goto 666
      else
      new_ind=0
      endif
666   ind_dec = new_ind*(pr-d(1))
      new_ind1 = hindir-new_ind*(pr-d(1))
      WRITE (8,*)'direct project cost increase
      is:',cost_increase
      WRITE (8,*)'indirect project cost decrease is:',ind_dec
      WRITE (8,*)'project cost increase is:', cost_increase-
      new_ind
      WRITE (8,*)'new direct project cost is:',
      cost_increase+dirtot
      WRITE (8,*)'new indirect project cost is:', new_ind1
      new_total=dirtot+cost_increase+new_ind1
      WRITE (8,*)'new total project cost is:', new_total
      open (11,file=' projco.dat',status='old')
      read (11,*) old_total
      close (11)
      open (7,file='new.dat',status='old')
      read (7,*) new_b
      close (7)

      if (new_b.eq.1) then
      open (10,file='X2.dat',status='old')
      WRITE (8,*)'activity to be accelerated:'
      do 83000 f=1,NN
```

```
if (f.gt.(32+aa)) then
gfa=f-32-aa
endif
read (10,*) EE(f)
if (EE(f).lt.d(f)) then
     if (gfa.eq.1) then
     WRITE (8,*)' excavate topsoil'
     endif
     if (gfa.eq.2) then
     WRITE (8,*)'excavate to reduce level'
     endif
     if (gfa.eq.3 .or. gfa.eq.4 .or. gfa.eq.5) then
     WRITE (8,*)'excavate/compact foundations'
     endif
     if (gfa.eq.6) then
     WRITE (8,*)' blinding concrete'
     endif
     if (gfa.eq.7) then
     WRITE (8,*)'formwork for foundations'
     endif
     if (gfa.eq.8) then
     WRITE (8,*)'reinforcement for foundations'
     endif
     if (gfa.eq.9) then
     WRITE (8,*)'concrete for foundations'
     endif
     if (gfa.eq.10) then
     WRITE (8,*)'roof asphalt'
     endif
     if (gfa.eq.11) then
     WRITE (8,*)'internal brickwork'
     endif
     if (gfa.eq.12) then
     WRITE (8,*)'internal blockwork'
     endif
     if (gfa.eq.13) then
     WRITE (8,*) 'external brickwork'
     endif
     if (gfa.eq.14) then
     WRITE (8,*)'external blockwork'
     endif
     if (gfa.eq.15) then
     WRITE (8,*)'internal finishes'
     endif
     if (gfa.eq.16) then
     WRITE (8,*)'wall finishes'
     endif
     if (gfa.eq.17) then
     WRITE (8,*)'floor finishes'
     endif
     if (gfa.eq.18) then
     WRITE (8,*) 'ceiling finishes'
     endif
     if (gfa.eq.19) then
```

```
      WRITE (8,*)'external finishes'
      endif
      if (gfa.eq.20) then
      WRITE (8,*) 'lift services'
      endif
      if (gfa.eq.21) then
      WRITE (8,*) 'mechanical services'
      endif
      if (gfa.eq.22) then
      WRITE (8,*) 'electrical services'
      endif
      if (gfa.eq.23) then
      WRITE (8,*)'formwork for ground floor slab/beam'
      endif
      if (gfa.eq.24) then
      WRITE (8,*)'reinforcement for ground floor slab/beam'
      endif
      if (gfa.eq.25) then
      WRITE (8,*)' concrete for ground floor slab/beam'
      endif
      if (gfa.eq.26) then
      WRITE (8,*)' gr-1st/ formwork for columns/walls'
      endif
      if (gfa.eq.27) then
WRITE (8,*)  'gr-1 / reinforcement for  columns/walls'
      endif
      if (gfa.eq.28) then
WRITE (8,*)'gr- 1 / reinforcement for columns/walls'
      endif
      if (gfa.eq.29) then
      WRITE (8,*)'gr-1 / formwork for staircases'
      endif
      if (gfa.eq.30) then
WRITE (8,*)' gr-1 / reinforcement for staircases'
      endif
      if (gfa.eq.31) then
      WRITE (8,*)' gr-1 / concrete for staircases'
      endif
      if (gfa.gt.31 .and. gfa.lt.(26+6*nofloor)) then
      do 8500 j= 1,(nofloor-1)
      tut=tut+1
             if (gfa.eq.(32+6*(j-1))) then
      WRITE (8,*) tut, '/ formwork for slab/beams'
           elseif (gfa.eq.(33+6*(j-1))) then
      WRITE (8,*)tut,'/ reinforcement for slab/beams'
           elseif (gfa.eq.(34+6*(j-1))) then
      WRITE (8,*) tut,'/ concrete for slab/beams'
           elseif (gfa.eq.(35+6*(j-1))) then
WRITE (8,*) tut,'-',tut+1, '/ formwork for columns/walls'
           elseif (gfa.eq.(36+6*(j-1))) then
WRITE (8,*) tut,'-',tut+1, '/ reinforcement for
columns/walls'
           elseif (gfa.eq.(37+6*(j-1))) then
```

```
      WRITE (8,*) tut,'-',tut+1,'/ reinforcement for
      columns/walls'
                endif
8500  continue
           endif
           if (gfa.eq.(26+6*nofloor)) then
           WRITE (8,*)'concrete for roof slab/beams'
           endif
           if (gfa.gt.(26+6*nofloor)) then
      do  86000  k=NN-3*(nofloor-2),NN,3
      lm=lm+1
      if (gfa.eq.k) then
      WRITE (8,*) lm,'-',lm+1, '/ formwork for staircases'
      endif
      if (gfa.eq.(k+1)) then
      WRITE (8,*) lm,'-',lm+1, '/ reinforcement for staircases'
      endif
      if (gfa.eq.(k+2)) then
      WRITE (8,*) lm,'-',lm+1, '/ concrete for staircases'
      endif
86000 continue
      endif
      endif
83000 continue
      endif
      close (10)
      open (10,file='X2.dat',status='old')
      do 8700 i=1,nn
      write (10,*) d(i),ee(i)
8700  continue
      close (10)


81    format (300f7.2/)
      if (old_total .lt. new_total) then
      WRITE (8,*) old_total, '-',d(1)+1, 'is the optimum '
      open (10,file='X.dat',status='old',form='formatted')
      do 81000 I=33+aa,71+aa
      write (10,81) d(i)
81000 continue
      close (10)
      open (11,file=' projco.dat',status='old')
      write(11,*) new_total
      write(11,*) cost_increase
      close (11)
      opt_dat=1
      endif
      if (d(1).lt. prm .or. d(1).gt.prm) then
      goto 3
      endif
      open (10,file='X1.dat',status='old',form='formatted')
      do 81002 I=33+aa,71+aa
      write (10,81) d(i)
81002 continue
```

```
      close (10)
      goto 3


21    do 211 jj=1,n
      a(1,jj)=0.
      do 212 i=2,m1
      k=nxi(i)
      a(1,jj)=a(1,jj)+icj(k)*a(i,jj)
212   continue
      a(1,jj)=a(1,jj)-icj(jj)
211   continue
      b(1)=0
      do 214 i=2,m1
      k=nxi(i)
      b(1)=b(1)+icj(k)*b(i)
214   continue
      zcm=a(1,1)
      jm=1

      do 109 jj=2,n
      if (kod .eq. 1) THEN
      goto 106
      ENDIF

      if ((a(1,jj)-zcm) .lt. 0) THEN
      goto 107
      else
      goto 109
      endif
106   if ((a(1,jj)-zcm) .le.0) THEN
      goto 109
      else
      goto 107
      endif
107   zcm=a(1,jj)
      jm=jj
109   continue
      ck=kod*zcm

      if(CK .ge.tol) THEN
      goto 131
      ENDIF
      do 124 i=2,m1
      k=nxi(i)
      if((icj(k) .gt.cv ).or.(icj(k).lt.cv)) THEN
      goto 124
      ENDIF
      if(b(i) .le. tol) THEN
      goto 124
      ENDIF
      open (2,file=' trial.dat',status='old')
      write (2,*) b(i)
      close (2)
```

```
      write(8,*) 'no feasible solution',k,b(i),i
      WRITE (8,*)'This project can not be accelerated more.
      write (8,*)'
      close (8)
124   continue
      npr=2
      goto 4444
131   xm=1000000000000000000.0

      im=0
      do 139 i=2,m1
      if(a(i,jm) .le. 0) then
      goto 139
      ENDIF
      xx=b(i)/a(i,jm)
      if ((xx-xm) .ge. 0) then
      goto 139
      ENDIF
      xm=xx
      im=i
139   continue
      if (im .gt.0) then
      goto 151
      ENDIF
      write(8,*) 'unbounded solution',jm
      close (8)
151   xx=a(im,jm)
      b(im)=b(im)/xx
      iter=iter+1

      do 152 jj=1,n
      a(im,jj)=a(im,jj)/xx
152   continue

      do 161 i=1, m1
      if ((i-im) .eq. 0) then
      goto 161
      ENDIF
      xx=a(i,jm)
      b(i)=b(i)-xx*b(im)


      do 154 jj=1,n
      a(i,jj) =a(i,jj)-xx*a(im,jj)
154   continue
161   continue

      nxi(im)=jm
      if(npr .eq. 1) then
      goto 4444
      ENDIF
      goto 21
```

```
3     open (11,file=' projco.dat',status='old')
      write(11,*) new_total
      write(11,*) cost_increase
      close (11)
      if (d(1).le.prm .or. d(1).gt.prm) then
      new_b = 1
      if (new_b.eq.1) then
      open (7,file='new.dat',status='old')
      write (7,*) new_b
      write (7,*) d(1)-1
      close (7)
      endif
      endif
      write (8,*)'
      write (8,*)'
      close (8)
7777  end
```